UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

SEGUIMIENTO AUTOMÁTICO DE OBJETIVOS CON DRONES MEDIANTE ALGORITMOS DE TRACKING.

Roi Rico Díaz.
Tutor: Fulgencio Navarro Fajardo.
Ponente: Jesús Bescós Cano.

Junio 2016

SEGUIMIENTO AUTOMÁTICO DE OBJETIVOS CON DRONES MEDIANTE ALGORITMOS DE TRACKING.

Roi Rico Díaz.
Tutor: Fulgencio Navarro Fajardo.
Ponente: Jesús Bescós Cano.

Video Processing and Understanding Lab
Departamento de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio 2016

Trabajo parcialmente financiado por el Ministerio de Economía y Competitividad del Gobierno de España bajo el proyecto TEC2014-53176-R (HAVideo) (2015-2017)

Resumen

El objetivo principal de este trabajo fin de grado es aprovechar la ventaja de usar cámaras móviles en lugar de cámaras fijas para tareas de videovigilancia como el seguimiento automático. Para ello, se ha propuesto modelar los problemas introducidos por la combinación de las condiciones de captura de las cámaras móviles con los retos a los que se enfrentan los algoritmos de seguimiento. Una vez definidos los problemas, se ha pasado a evaluar su efecto a nivel de resultados sobre los algoritmos de seguimiento del estado del arte mediante una etrategia de *toy-example* con un *dataset* propio y sintético. Modelados los efectos, se han propuesto dos modificaciones orientadas a la reducción de la borrosidad de la secuencias y a la estabilización de la cámara mediante puntos de interés. Dichas propuestas buscaban paliar los efectos de los problemas detectados. Para evaluar los efectos deseados de las mejoras se ha evaluado el sistema sobre el *dataset* sintético, y una vez corroborados se ha generado un *dataset* real a partir de secuencias grabadas con UAVs que permitiera evaluar la propuesta en un entorno real. Los resultados obtenidos permiten concluir que se ha logrado satisfacer el objetivo propuesto al comienzo de trabajo. Asi mismo, se considera que este proyecto puede ser el punto de partida de aplicaciones de visión sobre dispositivos de caputra a bordo de UAVs.

Palabras clave

Seguimiento, UAV, *jitter*, evaluación comparativa

Abstract

The main objective of this Final Degree Thesis is to take advantage of using mobile cameras instead of fixed cameras for video-surveillance tasks like tracking. For that purpose, problems introduced by combining the capture conditions of mobile cameras with the challenges faced by tracking algorithms are modeled. Once the problems have been defined, their effect the results of state of the art tracking algorithms is evaluated through a toy-example strategy with a proposed synthetic dataset. After modeling the aforementioned effects, two modifications oriented to a reduction of the blurring in the sequences and a stabilization of the camera by using points of interest have been proposed. These proposals sought to alleviate the effects of the detected problems. To check the desired effects of the improvements, the system has been evaluated on the synthetic dataset, and once these improvements have been corroborated, a real dataset with sequences captured by an UAV has been generated in order to evaluate the proposal in a real environment. The obtained results allow to conclude that the objective proposed at the beginning of this thesis has been accomplished with satisfaction. Additionaly, it has been considered that this project could be the starting point for vision applications on capture devices aboard UAVs.

Keywords

Tracking, UAV, *jitter*, *comparative evaluation*

Agradecimientos

Agradecer a familiares, amigos y allegados que han hecho que esto sea posible, así como a mi tutor Fulgencio Navarro Fajardo por la ayuda proporcionada desde el principio hasta el final del proyecto. Agradecer también a los compañeros de laboratorio, alumnos y profesores, por generar un gran ambiente de trabajo y por estar dispuestos en todo momento a ayudar en caso de requerirlo.

Índice general

Resumen v

Abstract vii

Agradecimientos ix

1. Introducción. 1
1.1. Motivación. 1
1.2. Objetivos. 1
1.3. Estructura de la memoria. 2

2. Estado del arte. 3
2.1. Estudio de tecnologías de UAV y tracking. 3
2.2. Retos de tracking. 5
2.3. Retos de UAVs. 8

3. Evaluación Comparativa y Toy Example. 11
3.1. Estudio y selección de algoritmos de seguimiento. 11
3.1.1. Cruce de retos. 11
3.1.2. Análisis de evaluaciones del estado del arte. 12
3.1.3. Selección de algoritmos de seguimiento. 13
3.2. Toy Example 14
3.2.1. Dataset 15
3.2.2. Métricas. 16
3.3. Resultados y análisis. 18
3.3.1. Retos de tracking. 20
3.3.2. Retos de UAVs. 22
3.4. Conclusiones. 23

4. Propuesta de mejora. 25
4.1. Visión general del sistema. 25
4.2. Mejoras introducidas 26
4.2.1. Mejora frente al efecto de blurring. 27
4.2.2. Mejora frente al efecto de jitter. 27
4.3. Evaluación comparativa en Toy example. 31

4.4. Conclusiones 33

5. Evaluación y resultados. **35**
5.1. Framework de evaluación. 35
5.1.1. Dataset. 35
5.1.2. Métricas. 37
5.2. Resultados y análisis. 37
5.3. Conclusiones 39

6. Conclusiones y trabajo futuro. **41**
6.1. Conclusiones. 41
6.2. Trabajo futuro. 42

Bibliografía **44**

A. Análisis teórico y práctico de algoritmos de tracking **51**

B. Resultados del toy example. **55**
B.1. CT. 55
B.2. DSST. 56
B.3. IVT 56
B.4. KCF 57
B.5. KLT 57
B.6. NCC 58
B.7. SAMF 58

C. Resultados de las secuencias reales. **59**
C.1. CT 59
C.2. DSST 60
C.3. IVT 60
C.4. KCF 61
C.5. KLT 61
C.6. NCC 62
C.7. SAMF 62

Índice de figuras

3.1. Gráficas de *F1-score* obtenidas para los retos de *tracking*. 19
3.2. Media de *F1-score* de todos los algoritmos por reto para los efectos de UAV. 22

4.1. Diagrama de bloques del sistema. 25
4.2. Diagrama de bloques del sistema con las correcciones añadidas. 26
4.3. Resultados de *F1-score* obtenidos por los algoritmos. 31
4.4. Comparativa de la media de *F1-score* de todos los algoritmos entre las secuencias corregidas y las originales para cada reto. 32

5.1. Resultados de *F1-score* obtenidos por los algoritmos. 38
5.2. Comparativa de la media de *F1-score* de todos los algoritmos entre las secuencias corregidas y las originales para cada reto. 39

A.1. Comparación entre los algoritmos del VOT para retos indicados (esquina superior izquierda). Fuente [1]. 54

Índice de tablas

3.1. Secuencias del *toy example* y número de *frames* de cada una. 15

5.1. Secuencias reales y número de *frames* de cada una. 36

B.1. Puntuaciones obtenidas por CT para las secuencias grabadas, generadas y corregidas del *toy example*. 55
B.2. Puntuaciones obtenidas por DSST para las secuencias grabadas, generadas y corregidas del *toy example*. 56
B.3. Puntuaciones obtenidas por IVT para las secuencias grabadas, generadas y corregidas del *toy example*. 56
B.4. Puntuaciones obtenidas por KCF para las secuencias grabadas, generadas y corregidas del *toy example*. 57
B.5. Puntuaciones obtenidas por KLT para las secuencias grabadas, generadas y corregidas del *toy example*. 57
B.6. Puntuaciones obtenidas por NCC para las secuencias grabadas, generadas y corregidas del *toy example*. 58
B.7. Puntuaciones obtenidas por SAMF para las secuencias grabadas, generadas y corregidas del *toy example*. 58

C.1. Puntuaciones obtenidas por CT para las secuencias reales grabadas y corregidas. 59
C.2. Puntuaciones obtenidas por DSST para las secuencias reales grabadas y corregidas. 60
C.3. Puntuaciones obtenidas por IVT para las secuencias reales grabadas y corregidas. 60
C.4. Puntuaciones obtenidas por KCF para las secuencias reales grabadas y corregidas. 61
C.5. Puntuaciones obtenidas por KLT para las secuencias reales grabadas y corregidas. 61

C.6. Puntuaciones obtenidas por NCC para las secuencias reales grabadas y corregidas. 62
C.7. Puntuaciones obtenidas por SAMF para las secuencias reales grabadas y corregidas. 62

Capítulo 1

Introducción.

1.1. Motivación.

Dentro del ámbito de la videovigilancia, y concretamente de tareas de seguimiento de objetivos, la mayoría de los sistemas de captura están compuestos por dispositivos fijos. Este tipo de dispositivos son sencillos de integrar y utilizar, y su coste es bastante reducido. Adicionalmente, la carencia de movimiento del dispositivo crea unas condiciones de captura muy favorables.

Sin embargo, estos dispositivos presentan un área de visión muy limitado. Por ello, para garantizar la cobertura de grandes áreas se requieren soluciones menos beneficiosas. Algunas soluciones consisten en incrementar el número de dispositivos o elevar su ubicación para cubrir más área, incrementando los costes o empeorando las condiciones de captura. Este problema es extensible al funcionamiento de algoritmos a largo plazo en tareas como el seguimiento, donde el objetivo se mueve libremente y se sale pronto del área de visión de la cámara.

Es en la actualidad donde el foco comienza a centrarse en otros sistemas de captura como las redes de cámaras, que requieren interacción entre las vistas, o las cámaras móviles, que empeoran notablemente las condiciones de captura. En conclusión, para garantizar el funcionamiento de tareas de videovigilancia, como el seguimiento de objetivos, en grandes áreas y/o a largo plazo, se requiere el cambio de tipos de dispositivos y algoritmos a utilizar.

1.2. Objetivos.

El principal objetivo del proyecto es garantizar el funcionamiento de algoritmos de videovigilancia, de seguimiento concretamente, a largo plazo y/o en grandes áreas.

La propuesta se centra por lo tanto en dos áreas. En primer lugar en el tipo de dispositivos de captura, que serán cámaras móviles a bordo de dispositivos voladores no tripulados (UAVs). En segundo lugar en algoritmos de seguimiento en condiciones de cámara fija , que requerirán su adaptación a las nuevas condiciones de captura para garantizar su funcionamiento.

Para ello, el objetivo principal del proyecto se ha afrontado desde la consecución de una serie de objetivos parciales:

- Estudio del estado del arte, tanto de los UAVs, para identificar retos del sistema de captura, como de los algoritmos de seguimiento, para garantizar el uso de las mejores aproximaciones en el sistema.
- Análisis de los efectos de las condiciones de captura de los UAV, generadas sintéticamente, sobre algoritmos de seguimiento para la posterior propuesta de mejoras. Implementación de las propuestas de mejora y corroboración de su funcionamiento sobre las secuencias sintéticas.
- Generación de un conjunto de datos real grabados con un UAV para la evaluación de la propuesta.

1.3. Estructura de la memoria.

La memoria se dividirá en los siguientes capítulos:

- Capítulo 1. Introducción: introducción, motivación y objetivos del proyecto.
- Capítulo 2. Estado del arte: estudio de las tecnologías actuales de UAVs y de *tracking,* y de los retos debidos a ambos en conjunto.
- Capítulo 3. Evaluación comparativa y *toy example*: donde se realizará una preselección de algoritmos, se presentará el *dataset* grabado para el *toy example* y se evaluará y analizará el rendimiento de los algoritmos.
- Capítulo 4. Propuesta de mejora: donde se mostrarán las correcciones de *blurring* y de *jitter* y se evaluará y analizará el rendimiento de los algoritmos sobre el *dataset* corregido.
- Capítulo 5. Evaluación y resultados: donde se evaluará y analizará con y sin correcciones el rendimiento de los algoritmos para el *dataset* real.
- Capítulo 6. Conclusiones y trabajo futuro.
- Referencias y anexos.

Capítulo 2

Estado del arte.

2.1. Estudio de tecnologías de UAV y tracking.

La tecnología de los drones o UAVs (*Unmanned Aerial Vehicles*) se lleva investigando desde 1916 [2] principalmente para uso militar. Sin embargo, ha sido hace relativamente poco cuando se ha empezado a usar en ámbitos más comerciales y por lo tanto más accesibles al público general [3].

Su aspecto más característico, como su propio nombre indica, es que se trata de vehículos voladores no tripulados, esto es, que no necesitan de un piloto humano controlando el aparato a bordo. Otra de sus características principales es su reducido tamaño, al no tener, por lo general, que llevar carga en el interior. Si bien es cierto que existen drones para transporte, y por lo tanto de gran tamaño, suelen estar restringidos a ámbitos militares.

Su reducido tamaño deriva a su vez en que los gastos tanto de producción como de manutención sean reducidos. A su vez, los requisitos de confort o seguridad que deben presentar aparatos destinados al transporte de personas son obviados, pudiéndose enfrentar condiciones de navegación más adversas. Un claro ejemplo es su uso en conflictos bélicos [4].

Todo ello ha contribuido a la reciente popularización de esta tecnología, y la sitúa como una de las más crecientes y con más potencial del mercado.

El manejo de estos aparatos permite llevar a cabo una clasificación de los mismos. Las dos grandes categorías son: los UAV pilotados por control remoto [5] y los UAV auto guiados [6] o *self-guided*.

Dentro del conjunto de los UAV *self-guided* también es posible realizar una clasificación en función de cómo enfrentan los retos que supone el auto guiado, como puede ser el *crash avoiding* [7, 8, 9, 10, 11]. Algunos UAVs utilizan información recogida de

diversos sensores como pueden ser cámaras [12, 13, 14], LIDARs [14], radares [15], así como otro tipo de sensores. Otras soluciones de auto guiado se basan en el control empleando datos de geolocalización y posicionamiento [3, 4, 16].

Por otra parte, en la línea de las aplicaciones para las que estos aparatos pueden ser de utilidad destacan todas aquellas relacionadas con la obtención de imágenes. En aquellas tareas en las que previamente resultaba imposible o muy costosa la extracción de imágenes por la posición, la movilidad de la cámara requerida o la accesibilidad de la zona, la inclusión de mecanismos de captura de imágenes en los UAVs se ha postulado como la mejor solución. Además, la posibilidad que ofrecen algunos UAVs de transmitir la información visual que están recibiendo en tiempo real [17] amplía aún más el rango de aplicaciones en las que los drones pueden resultar de utilidad.

En el ámbito comercial se pueden destacar las aplicaciones para la monitorización del tráfico, como por ejemplo el control de accidentes [12, 18] o la medición de densidad de tráfico [17]. Las aplicaciones en casos de emergencias son también muy relevantes, destacando aquellas para la detección, localización y monitorización de incendios [19, 20, 21], la monitorización de zonas catastróficas [22, 23], o la búsqueda y localización de humanos perdidos en zonas de difícil acceso como pueden ser desiertos o montañas de gran altitud [24]. Por último, y no menos importantes, las aplicaciones de videovigilancia, por ejemplo para observación y monitorización de grandes zonas de terreno [4], o monitorización móvil de zonas restringidas [25].

En el ámbito militar se emplean por ejemplo para realizar misiones de espionaje [4] o como plataforma de disparo de misiles [4, 26].

Muchas de estas aplicaciones, incluidas las propias técnicas usadas para el auto guiado, que se llevan a cabo con cámaras, se basan en algoritmos de procesamiento de vídeo. Unas de las técnicas más utilizadas en estas aplicaciones son las relacionadas con el concepto de *tracking* o seguimiento automático de objetivos en secuencias de vídeo. El *video tracking* es la capacidad de estimar la posición de uno u múltiples objetos de interés en la imagen y seguirlos en el tiempo, es decir, en la secuencia de imágenes posteriores a su detección/inicialización. El *tracking* por sí mismo es uno de los ámbitos de procesamiento de señales de vídeo en los que más se está trabajando en la comunidad científica (número de entradas de *Google Scholar*: 2'840'000), ya que proporciona una base de información que puede ser muy útil para diferentes aplicaciones de nivel superior.

Cabe destacar que en la actualidad, los algoritmos de *video tracking* se aplican mayoritariamente en cámaras estáticas, las cuales a pesar de presentar condiciones de análisis más sencillas, limitan el campo de visión reduciendo de este modo la capacidad del algoritmo de seguir al objetivo a largo plazo o en grandes áreas. La posibilidad

de montar cámaras en vehículos como puedan ser los UAVs ofrece una ampliación del escenario de aplicación de las técnicas de *tracking*. Un claro ejemplo son los sistemas de videovigilancia que tratan de monitorizar usuarios en áreas amplias y complejas, que a priori necesitan varias cámaras, y en [27, 28] solventan mediante una única cámara móvil en un UAV.

Otras aplicaciones del *tracking* en UAVs fuera de la videovigilancia son por ejemplo las citadas anteriormente, la búsqueda y localización de humanos perdidos en zonas remotas de difícil acceso [24], o la monitorización del tráfico [12, 17, 18].

Como contrapartida, el desarrollo del *tracking* para vídeo con condiciones como las que se tienen a bordo de drones está mucho menos generalizado, y las aproximaciones usadas resultan ad-hoc para escenarios y situaciones definidos.

Se puede concluir que el uso de UAVs con cámaras tiene un potencial de aplicación enorme y que por lo tanto, el correcto funcionamiento de los algoritmos de procesamiento de vídeo, y más concretamente los de *video tracking*, en estos entornos cobra una gran relevancia. Sin embargo, es necesario un trabajo de investigación en profundidad orientado al análisis de las características que deben de presentar los algoritmos de *tracking* que se vayan a emplear en estas condiciones. El objetivo de este estado del arte será recopilar y analizar las propiedades, tanto de la plataforma de captura, esto es, las cámaras de los UAVs, como de los *trackers*, y seleccionar aquellos que mejor se adapten para su posterior evaluación.

2.2. Retos de tracking.

Se han definido los algoritmos de *video tracking* como el conjunto de técnicas utilizadas para estimar la posición de uno u múltiples objetivos en la escena a lo largo del tiempo. Existen numerosas y muy diversas aproximaciones del estado del arte. Para poder evaluar los distintos retos a los que se enfrenta el *tracking*, se va a presentar la división general [29] de las estrategias principales seguidas en el estado del arte: los algoritmos discriminativos y los algoritmos de *matching*.

Los algoritmos discriminativos generan un modelo diferenciando entre el objetivo en primer plano y el fondo. Se emplea para ello un clasificador, el cual distingue entre los píxeles del objetivo y los píxeles del fondo, que se actualiza según le llegan nuevas muestras.

Por su parte los algoritmos basados en *matching* realizan un modelo del objetivo y lo van buscando en las sucesivas imágenes.

Todas estas estrategias se presentan para funcionar en entornos con un conjunto de retos a los que el algoritmo debe ser robusto. Dichos retos pueden referirse a la

escena, al objetivo de seguimiento o incluso al propio mecanismo de captura.

Debido a que la categorización de los retos de *tracking* no es homogénea, se ha recurrido al libro [30], referente del estado del arte de la técnica, para presentar una clasificación de los mismos lo más fiable y amplia posible. Esta clasificación de retos se dividirá en aquellos que únicamente tienen que ver con los objetivos de seguimiento y aquellos que están relacionados con cambios en la escena.

Los cambios en los objetivos, o cambios en la superficie de interés pueden dar lugar a resultados erróneos en el seguimiento. En las estrategias de *matching*, donde se suele tener un modelo del frente u objetivo, la robustez frente a estos cambios se convierte en un aspecto crítico. Por su parte algunas de las estrategias de *tracking* discriminativo como las que aparecen en [29], basadas en incluir modelos de fondo locales, presentan cierto éxito en este aspecto pero a costa de verse influidos de manera más severa por los retos que afectan a la escena. Algunos de los posibles cambios que se engloban dentro de esta categoría son:

- Cambios de escala: Cualquier objeto que se mueva tanto hacia la cámara como en contrasentido afectará al tamaño del mismo dentro de la escena, por lo que el algoritmo de *tracking* tiene que ser robusto a estos cambios de escala del objetivo.

- Cambios de información (color, nivel de gris): Cambios de color o nivel de gris en la superficie del objetivo. Puede deberse a que el objetivo se ha girado con respecto a la cámara o a que se ha puesto ropa de otro color por encima.

- Cambios de aspecto: Cambios en la forma del objetivo, como por ejemplo cuando una persona extiende los brazos o se agacha.

Por otra parte, se presentan los retos relacionados con la escena. Afectan a ambas estrategias, pero pueden llegar a ser más críticos en los modelos discriminativos, donde se trata de clasificar frente y fondo, y por lo tanto cambios en el segundo pueden dar lugar a resultados erróneos. En este grupo se encuentran dos tipos principales de retos: los relacionados con la complejidad de la escena y los relacionados con cambios en la propia escena.

En cuanto a la complejidad de la escena, los tipos de eventos que más penalizan a los algoritmos de *tracking* son los camuflajes y las oclusiones parciales o totales del objetivo. Las causas de estos eventos se engloban en:

- El *clutter* o aglomeración de elementos potencialmente distractores para el algoritmo. Se entiende por camuflaje, aquellas situaciones en las que la escena presente propiedades similares a las del frente, pudiendo dar lugar a confusión

entre el frente y el fondo. Dentro de los camuflajes, existen camuflajes por color, por aspecto, o incluso por movimiento, porque el movimiento sea similar al del objetivo. Los escenarios o áreas de la escena con *clutter*, esto es, con grandes acumulaciones de elementos distractores, darán lugar a más errores en los resultados de los algoritmos de seguimiento.

- Oclusiones: La desaparición del objetivo del seguimiento de la escena, o de parte del mismo, supone una dificultad añadida a la tarea de seguimiento. Aquellos *trackers* que utilicen modelos del objetivo de seguimiento serán más robustos a estas situaciones, sin embargo, si las oclusiones son prolongadas, las actualizaciones del modelo darán lugar a errores igualmente. En cualquier caso, aquellas escenas concurridas, o con numerosas zonas de oclusión tendrán un mayor nivel de complejidad que aquellas diáfanas.

Por último, los cambios en la información de la escena también añadirán dificultad al seguimiento. Estos cambios afectarán a priori a aquellos algoritmos que modelen la escena. Sin embargo, cambios en la escena pueden dar lugar a camuflajes, o a cambios en la apariencia del objetivo, por lo que no se pueden acotar los algoritmos afectados por este tipo de dificultades. Los dos eventos principales incluidos en esta categoría son los cambios de iluminación y el ruido.

- Cambios de iluminación: Cualquier cambio en la iluminación de la escena puede afectar tanto a un posible modelo del fondo como a la apariencia del objeto de interés en la misma. Según la naturaleza de la escena, p. ej. interiores o exteriores, los cambios de iluminación serán más locales o más generales, y de mayor o menor brusquedad. En cualquier caso, los cambios de iluminación generan unas dificultades notables y frecuentes, y por ello la mayoría de los algoritmos presentan estrategias para hacerles frente.

- El ruido de la escena está frecuentemente relacionado con la calidad del sistema de captura. Bajas resoluciones, poco contraste o altos niveles de compresión generan a menudo imágenes ruidosas. Aquellos algoritmos que requieran extraer información precisa de la imagen, véanse texturas o gradientes, se verán notablemente afectados por estos efectos.

Una vez llevada a cabo una recopilación de los diversos problemas o retos a los que se deben enfrentar los algoritmos de seguimiento, se va a proceder a estudiar la naturaleza de las imágenes o secuencias capturadas con UAVs, con el fin de poder evaluar los algoritmos de *tracking* frente a sus problemas propios y a los derivados de la naturaleza de su entorno de aplicación.

2.3. Retos de UAVs.

Incluir una o varias cámaras a bordo de un UAV resulta muy útil para diferentes aplicaciones como hemos explicado anteriormente. Sin embargo no resulta sencillo, puesto a que habrá que tener en cuenta los factores derivados del entorno de captura y su efecto en las imágenes que posteriormente serán tratadas por los algoritmos de *Computer Vision*.

Por ello el tratamiento de secuencias de imágenes captadas con cámaras a bordo de UAVs presenta una serie de retos, los cuales clasificaremos en dos tipos: los debidos únicamente al movimiento del UAV y los debidos a las características de la cámara. Dentro del primer tipo de retos se encuentran:

- *Jitter* en las imágenes: La cámara situada a bordo del UAV tiene una vibración debida a la baja estabilidad del propio UAV. Si dicha vibración es perceptible en la imagen, tareas relativamente sencillas para el *tracker*, como puedan ser la estimación del movimiento del objetivo, resultan notablemente complejas [31]. En este ámbito, y orientado a UAVs, se han desarrollado diversos algoritmos, como [32], los cuales buscan estabilizar y rectificar la secuencia de imágenes para su posterior uso.

- Balanceo o *swaying* en las imágenes: El efecto conocido como *swaying* se debe al movimiento homogéneo de diversos objetos en la imagen, provocando una falsa ilusión de movimiento de la cámara. El *swaying* o balanceo coge su nombre del movimiento que tienen los árboles o plantas debido al viento, ejemplo más común de *swaying*. En este ejemplo las ramas de los árboles o plantas se moverían en la misma dirección dependiendo de la dirección y fuerza del viento. Al tratarse de un movimiento homogéneo dentro de la secuencia de imágenes la cámara entiende que dicho movimiento es debido a su propio movimiento, lo cual es un error, y trata de compensarlo [33]. El *swaying* puede provocar la oclusión de alguno de los objetos presentes en la secuencia de imágenes si se encuentran cerca de los bordes de las imágenes [31]. Como en el *jitter*, se solucionará dicho problema filtrando la secuencia de imágenes una a una para estabilizarlas. Los efectos de *swaying* y *jitter* son fáciles de paliar si aparecen por separado, sin embargo suelen aparecer juntos cuando se trata de cámaras a bordo de UAVs pequeños, lo cual complica la estabilización y rectificación de la imagen.

- Movimiento de la cámara: A la hora de realizar el seguimiento de cualquier objeto en una secuencia de imágenes resulta común utilizar el movimiento de la cámara, puesto a que ayuda a comprender el movimiento ocurrido en la escena.

Existen diferentes modos de hallar el movimiento de la cámara. La información puede ser extraída de sensores a bordo del UAV o de la propia imagen. Para calcular el movimiento de la cámara a partir de la imagen se emplean algoritmos como [14], el cual propone combinar características de los puntos estáticos en la escena con el flujo óptico de la imagen. Si los cambios son muy grandes o bruscos entre imagen e imagen será difícil compensar el movimiento de la cámara.

- Cambios de iluminación: En caso de trabajar con cámaras fijas los cambios de iluminación que puedan ocurrir sólo son debidos al paso del día o a alguna nube. Dichas cámaras han de ser robustas a estos cambios. Las cámaras móviles necesitan un grado de robustez mucho mayor que las fijas, y es que una cámara montada a bordo de un UAV tiene que soportar muchos más cambios de iluminación y en menos tiempo. Estos cambios de iluminación pueden ser causados por pasar de una zona bien iluminada a una zona con sombra o por pasar de enfocar al suelo a enfocar al cielo, además de otras causas.

El movimiento del UAV puede hacer que la secuencia de imágenes también se vea afectada por las características que tiene la propia cámara. De entre ellas destacamos la resolución, el tiempo de exposición y el ratio de cuadros por segundo. Las últimas dos características se ven más afectadas por los movimientos bruscos producidos por un UAV.

- Resolución: Las cámaras empleadas a bordo de UAVs son generalmente cámaras de una calidad media, debido a que no es rentable montar cámaras de alta calidad. Por ello, la resolución de las imágenes captadas a una cierta distancia no se verán nítidas, pudiendo afectar de este modo a la detección de un objetivo.

- Tiempo de exposición: Cuando se trabaja en condiciones de baja iluminación se aumenta el tiempo de exposición de la cámara con el fin de que entre más luz en el sensor y por lo tanto salgan imágenes más claras. Esto funciona bien en ausencia de movimiento, sin embargo al estar la cámara montada en un UAV hay que contar con el movimiento en ocasiones brusco del UAV. Las imágenes dejarán de ser nítidas, puesto a que al permanecer mayor tiempo abierto el obturador cualquier objeto que se mueva en la imagen aparecerá borroso.

- Cuadros por segundo: La baja calidad de la cámara o del sistema de transmisión puede hacer que el número de cuadros por segundo no sea el suficiente, provocando de este modo una ilusión falsa de movimiento o movimientos inesperados de los objetos en la imagen. Esto puede confundir al *tracker* a la hora de estimar la trayectoria y velocidad de un objetivo.

Capítulo 3

Evaluación Comparativa y Toy Example.

3.1. Estudio y selección de algoritmos de seguimiento.

3.1.1. Cruce de retos.

Una vez hemos presentado los retos de los algoritmos de *tracking* y de los UAVs por separado, el próximo paso será averiguar cuáles de estos retos tendrían un mayor efecto a la hora de montar un algoritmo de seguimiento en un UAV. Para ello, tenemos que tener en cuenta los dos factores más importantes relativos a la naturaleza de los UAVs: el movimiento generalmente brusco y la calidad de la/s cámara/s abordo. En este apartado no aparecen algunos de los retos mencionados en el estado del arte, debido a que su efecto resulta de menor relevancia que los aquí expuestos.

Muchos de los retos que aparecen al emplear una cámara a bordo de un UAV, como puedan ser la resolución de la cámara, etc., son impuestos por las características de la cámara, por lo que no tendrán un efecto especialmente importante a la hora de llevar a cabo este proyecto. No obstante, el movimiento de la cámara y el *jitter* sí que resultan factores diferenciadores a la hora de seguir un objeto mediante un UAV con cámara a bordo. Y es que, el movimiento de la cámara se emplea, como ya explicamos en el estado del arte, para comprender el movimiento que se ha producido en la escena. Esta característica es de gran ayuda si la diferencia en cuestiones de movimiento entre una imagen y la siguiente es reducida. Sin embargo, no sirve de nada si la diferencia es grande, lo cual en muchos casos puede ser un problema si se graba desde un UAV. Por otra parte si se añade a este inconveniente el *jitter* el problema aumenta, debido a que si éste es muy notable será imposible realizar una reconstrucción del movimiento de la cámara y por lo tanto el algoritmo de *tracking* perderá a su objetivo.

De entre los retos relativos al *tracking*, los más potenciados debido al movimiento del UAV son los cambios de escala y de aspecto. Esto se debe a que el UAV tiene la capacidad de moverse en cualquier dirección y con una velocidad que no tiene que ser la del objetivo, por lo que el UAV se acercará, se alejará (cambios de escala) y se podrá mover libremente alrededor del objetivo (cambios de aspecto). Además de estos hay otros retos presentes en el *tracking* que serán relevantes a la hora de llevar a cabo este proyecto. Hablamos de las oclusiones y del *clutter*. El primero de estos retos afecta con mayor asiduidad que los presentados con anterioridad. Esto se debe a que el algoritmo de *tracking* en un escenario real no tiene conocimiento de la escena en la que se mueve el UAV, por lo que no es capaz de predecir los objetos que se puedan interponer entre la cámara y el objetivo. A su vez el movimiento en un UAV es prácticamente continuo, por lo que el algoritmo tendrá que lidiar continuamente con oclusiones. Estas dos características también afectarán al *clutter*, sin embargo este reto puede tener un efecto menor en algunos escenarios debido al movimiento del UAV, por lo que su impacto será menor que el de las oclusiones.

Por último, el cambio de iluminación. Este reto es común tanto para el *tracking* como para los UAVs. En el caso del *tracking* el reto surge a la hora de poder manejar diferentes ambientes de iluminación en la misma escena, mientras que en los UAVs el problema se reduce a que éste puede filmar áreas con iluminaciones muy distintas en corto espacio de tiempo. Este reto es uno de los más importantes con los que hay que trabajar, debido a que puede aparecer tanto por el movimiento del UAV como por la diferente luz natural en la escena.

3.1.2. Análisis de evaluaciones del estado del arte.

Uno de los elementos indispensables a la hora de realizar este proyecto es el algoritmo de *tracking*. Durante la última década se han publicado muchas investigaciones en el ámbito del seguimiento de objetos, llegando a publicarse alrededor de cuarenta investigaciones anuales en las conferencias de mayor prestigio, como ECCV, ICCV y CVPR.

A diferencia de otros campos de *Computer Vision* como puedan ser el flujo óptico [34] o la segmentación [35], las investigaciones llevadas a cabo en el ámbito del *tracking* no suelen hacer uso de sets de datos estandarizados ni de un método de evaluación común para dichas investigaciones. Es por ello que resulta complicado comparar el rendimiento de los diferentes algoritmos de *tracking*, lo cual sería muy ventajoso para averiguar cuan eficientes son los nuevos algoritmos y observar cómo se van desarrollando los ya existentes.

Debido a la carencia de una metodología de evaluación común, se propusieron

numerosas iniciativas [29, 36, 37] con el fin de procurar a los investigadores una base común para evaluar el rendimiento de los *trackers*. De entre todos ellos, el de más relevancia en la actualidad es el conocido *Visual Objective Tracking Challenge* (VOT)[1]. Este proyecto nace en el año 2013 y a diferencia del resto, su intención es que los experimentos sean llevados a cabo por los propios creadores de los algoritmos. Para ello se provee al investigador de un *kit* de evaluación, en el cual tendrá que integrar su algoritmo, un set de datos completo y anotado y una metodología concreta de evaluación. Los resultados obtenidos serán mandados a un comité de evaluación para su análisis y posteriormente se colgarán los resultados definitivos en la página principal del VOT. Este reto se celebra todos los años, puesto a que esta tecnología avanza de forma rápida y los resultados pueden mejorar de forma significativa de año en año.

3.1.3. Selección de algoritmos de seguimiento.

Durante la realización de este proyecto vamos a proceder a comparar diferentes algoritmos elegidos previamente sobre diferentes escenarios. Es por ello que resulta de gran ayuda basar dicha elección en los resultados obtenidos por los diferentes algoritmos en uno de los *frameworks* de evaluación presentados con anterioridad. Las publicaciones más aceptadas hasta la fecha en relación a la evaluación de algoritmos de seguimiento de un solo objetivo con una sola cámara son el *Online Tracking Benchmark* (OTB) de Wu et al. [37], el estudio experimental de Smeulders et al. [29] y el *Visual Object Tracking* VOT 2014 *Challenge Results* [1].

OTB está orientado a la inicialización del objetivo, puesto a que es un problema muy frecuente, pero pocas veces afrontado. Para ello perturba espacial- y temporalmente el estado inicial del *bounding box* del *ground-truth*. A diferencia del OTB, [29] y [1] sí ofrecen los datos exactos de inicialización del *bounding box* en el primer *frame*, puesto a que ambas evaluaciones están más orientadas a observar el comportamiento de cada algoritmo ante una determinada escena en la que pueden aparecer diferentes retos a los que hacer frente. Este enfoque no es el que más interesa para este proyecto, por lo que el OTB queda excluido como referencia.

Tanto [29], como [1] evitan el uso de información extraída de los *frames* anteriores y posteriores del vídeo, así como la información obtenida a través de datos de entrenamiento (*offline training*). El único parámetro conocido para el algoritmo ha de ser su *bounding box* en el primer *frame* del vídeo. Ambos *benchmarks* evalúan prácticamente los mismos *trackers* con métodos objetivos y en cierto modo similares, sin embargo existen diferencias entre los dos. En primer lugar [1] cuenta con un número más elevado de *trackers* evaluados, un gran punto a favor para este *benchmark*

[1] `http://www.votchallenge.net/vot2014/`

puesto a que interesa que el rango de *trackers* a elegir sea cuanto más amplio mejor. Además de esta gran ventaja, también cuenta con un set de datos más extenso y cuidadosamente elegido y procesado. Mientras que [29] se basa en la librería de datos *Amsterdam Library of Ordinary Videos for tracking* ALOV++ para obtener un total de 315 secuencias de vídeo categorizadas según su longitud y/o contenido, [1] propone emplear secuencias de vídeo de ALOV, de OTB, del *benchmark* presentado por VOT en el 2013 [38], así como otras secuencias no publicadas hasta la fecha, para hacer un total de 394 secuencias previo procesado.

VOT2014 realiza un pre-procesado de las 394 secuencias con las que cuenta inicialmente. Dicho pre-procesado consiste en eliminar manualmente las secuencias en las que el número de *frames* sea menor de 200, las secuencias en tonos de grises, las secuencias que contienen objetivos no muy definidos (p.ej. fuegos de artificio) y las secuencias que contienen cortes. Los *frames* de cada secuencia han sido etiquetados manualmente de uno en uno mediante cinco atributos visuales que representan a cada reto presente en dicho *frame*: (i) oclusión, (ii) cambios de iluminación, (iii) cambios en el movimiento del objetivo, (iv) cambios de tamaño, (v) movimiento de la cámara. Muchos de estos *frames* no se corresponden con ninguno de los atributos anteriormente descritos, por lo que son etiquetados como (vi) neutrales.

Estas características que presenta [1] frente a las otras publicaciones presentadas con anterioridad hacen que sea el elegido como referencia para llevar a cabo este proyecto. De entre todos los *trackers* presentes en [1] se elegirán siete diferentes, basándose en las calificaciones obtenidas y en su relevancia hasta la fecha. Es por ello que a priori hay *trackers* que funcionarán mejor que otros y que, por lo tanto, obtengan mejores calificaciones. Sin embargo algunos de estos *trackers* están más extendidos y se emplean con más asiduidad, por lo que hemos considerado incluirlos en este proyecto a pesar de que sus resultados están lejos de ser los mejores.

Los *trackers* elegidos son los siguientes: DSST, SAMF, KCF, IVT, CT, NCC y KLT. Todos ellos han sido analizados en el Anexo A, para su posterior incorporación a la fase de evaluación. Junto con ellos, se ha incorporado una gráfica resumen de la evaluación realizada en [1], que ha sido la base de la selección de dichos algoritmos.

3.2. Toy Example

Para poder probar los diferentes algoritmos y realizar las mejoras pertinentes se ha realizado un *toy example* con el fin de modelar de manera muy simplificada los retos a los que tiene que enfrentarse el algoritmo de *tracking* en caso de estar montado sobre un dron. El objetivo por lo tanto es seleccionar los algoritmos que se

emplearán a la hora de trabajar con secuencias grabadas desde un UAV, así como evaluar posteriormente sobre ellos las futuras mejoras introducidas en determinadas secuencias para comprobar si pueden ser aplicables en las secuencias grabadas con un UAV.

3.2.1. Dataset

Las grabaciones llevadas a cabo se han realizado empleando una cámara fija situada a una altura de unos cuatro metros sobre el objetivo, modelando de esta manera como grabaría un dron. Al ser la cámara fija todo el movimiento presente en la escena, así como los retos que se han mostrado, han sido realizados por el objetivo, en estas secuencias una persona. Se grabaron siete escenas de las cuales seis representan diferentes retos para el *tracker*: cambios de aspecto, cambios de apariencia, oclusiones, camuflaje, movimiento de bote rápido y movimiento de bote lento. La otra secuencia restante consiste en una persona andando a través del rango de visión de la cámara. Todas las secuencias están grabadas independientemente de forma que cada una de ellas representa un reto por separado.

En la Tabla 3.1 se puede leer el número de *frames* que tiene cada secuencia correspondiente a un reto.

Retos	Nº de frames
Bounc_Mov_Lento	552
Bounc_Mov_Rapido	348
Cambio_Apariencia	476
Cambio_Aspecto	647
Camuflaje	689
Oclusiones	750
Paseo_Simple	277

Tabla 3.1: Secuencias del *toy example* y número de *frames* de cada una.

El número total de *frames* para todas las secuencias es 3'739. Todas las secuencias han sido grabadas en formato AVI (*Audio Video Interleave*) con un ratio de treinta cuadros por segundo y una resolución de 960x540 píxeles.

La secuencia más corta representa un paseo normal delante de la cámara. La más larga simula el reto de las oclusiones, en la que se hace pasar a una persona varias veces por delante de la persona objetivo con el fin de ocultar parte de su cuerpo. Dichas oclusiones han sido en su totalidad parciales. El resto de retos se han representado empleando una sola persona en la grabación. En los movimientos de bote la persona anda simulando el movimiento que tiene un balón botando, lo cual resulta muy poco

predecible para el algoritmo. La secuencia de cambios de aspecto se ha simulado andando el objetivo hasta el centro del rango de visión de la cámara y una vez ahí levantando los brazos y colocándolos en distintas posiciones. El reto de cambios de apariencia se ha realizado de modo parecido, sin embargo una vez se encontraba el objetivo en el centro de la escena, éste ha procedido a quitarse el jersey, cambiando de este modo el color y la forma del objetivo. Y por último el reto del camuflaje, en el cual se ha procedido del mismo modo que en el anterior, sin embargo en vez de quitarse el jersey esta vez el objetivo se ha puesto un jersey de color similar al color del suelo, puesto a que al estar grabando desde arriba el objetivo solo se podrá camuflar con el suelo.

A partir de las secuencias grabadas se han generado otras secuencias simulando algunos de los retos que no se han tenido en cuenta en las secuencias anteriores. De este modo los retos que se han cubierto generando estas secuencias son la presencia de ruido aleatorio en las imágenes, los cambios de iluminación en la escena y el *jitter*. El ruido aleatorio se ha simulado empleando una función que genera una matriz de valores aleatorios, y realizando diversos ajustes para que ésta se ajustase a la secuencia de forma realista. Para los cambios de iluminación se ha procedido a aumentar y disminuir de *frame* en *frame* el valor en nivel de grises de los píxeles, de modo que la secuencia fuera aumentando o disminuyendo la iluminación con el paso de los *frames*. Con el fin de simular el efecto de *jitter* provocado por la inestabilidad del UAV, se ha usado la misma función que para el ruido aleatorio dándole en esta ocasión unos valores máximos para mover de forma aleatoria la imagen dentro del rango de la cámara.

Una vez se han llevado a cabo las grabaciones se ha procedido a pre-procesarlas con el fin de poder anotarlas y emplearlas como set de datos para realizar evaluaciones de algoritmos de *tracking*. Dicho anotado ha sido llevado a cabo por una única persona, anotando todos los *frames* presentes en cada secuencia mediante la herramienta de anotación *VIA*. Posteriormente se ha extraído el archivo con el *ground-truth* de cada secuencia y se ha procedido a procesarlo con el fin de dejar en cada línea del archivo los datos de comienzo del *bounding box* (píxel de más arriba y píxel de mas a la izquierda del box) y su anchura y altura en píxeles. Se ha realizado este anotado tanto para las secuencias grabadas como para las secuencias en las que el *jitter* simulado ha sido corregido.

3.2.2. Métricas.

La métrica empleada para evaluar los algoritmos es la conocida como *F1-score* [29, 39]. Esta métrica deriva de *F-score*, la cual a su vez proviene del criterio de

PASCAL para la comparación de resultados. Todas estas métricas se basan en el solapamiento entre el *bounding box* obtenido por el algoritmo y el de las anotaciones para la evaluación.

Tanto *F-score* como el método de solape de PASCAL sugieren establecer un límite por encima del cual se considerará correcto el resultado obtenido para esa secuencia. En el criterio de PASCAL 3.1 se tiene que dar la condición de que el valor absoluto de la intersección dividido entre el de la unión de los píxeles del *bounding box* sea mayor o igual que 0,5. Este criterio solo es capaz de medir el solapamiento entre ambos *bounding boxes*, sin embargo no es capaz de diferenciar si el algoritmo está completamente perdido (falso negativo) o si el algoritmo está siguiendo a un objeto que no era el deseado (falso positivo). *F-score* tiene en cuenta esta circunstancia a la hora de puntuar el rendimiento del algoritmo para una determinada secuencia. Dentro de la ecuación que ha de cumplir *F-score 3.2* aparecen dos cualidades a medir del algoritmo: *precision* y *recall*. La primera de ellas consiste en dividir el número de verdaderos positivos (cuando el algoritmo sigue al objeto correcto) entre la suma del número de verdaderos positivos y falsos positivos. De este modo se podrá hallar la precisión del algoritmo para seguir a un objeto en particular sin perderse con otro similar. *Recall* consiste en dividir el número de verdaderos positivos entre la suma del número de verdaderos positivos y falsos negativos, obteniendo de este modo la robustez del algoritmo ante posibles pérdidas.

Las ecuaciones para el criterio de PASCAL y *F-score* respectivamente son:

$$\frac{|T^i \cap GT^i|}{|T^i \cup GT^i|} \geq 0{,}5, \tag{3.1}$$

donde T^i denota el *bounding box* obtenido por el *tracker* en el *frame i*.

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall}, \tag{3.2}$$

donde n_{tp}, n_{fp} y n_{fn} denotan el número de verdaderos positivos, falsos positivos y falsos negativos respectivamente en un vídeo.

F1-score engloba las dos métricas explicadas con anterioridad. Utiliza el criterio de PASCAL debido a que se trata de una métrica basada en el solapamiento entre el área del *bounding box* de los resultados obtenidos por el algoritmo y el área del *bounding box* de las anotaciones llevadas a cabo. Sin embargo, al igual que *F-score*, realiza medidas de *precision* y *recall*. La principal diferencia que introduce *F1-score 3.3* es que se trata de una métrica basada en el área. En este caso la precisión se mide dividiendo la unión entre el *bounding box* obtenido por el algoritmo y el extraído de

las anotaciones por el *bounding box* obtenido por el algoritmo, mientras que el *recall* se adquiere de dividir dicha unión entre el propio *ground-truth*. Después de obtener estos parámetros los multiplicamos y los dividimos entre su suma para luego multiplicarlos por dos. Esto se realiza para todas las *frames* presentes en la secuencia, por lo que todos los resultados obtenidos se suman para luego pesarlos entre el número de *frames* de la secuencia.

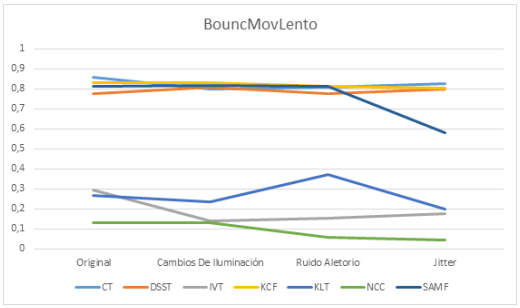
$$F1 = \frac{1}{N_{frames}} \sum_{i} 2 \cdot \frac{p^i \cdot r^i}{p^i + r^i}, \tag{3.3}$$

donde p^i y r^i representan respectivamente la precisión y el *recall* para el *frame* i.

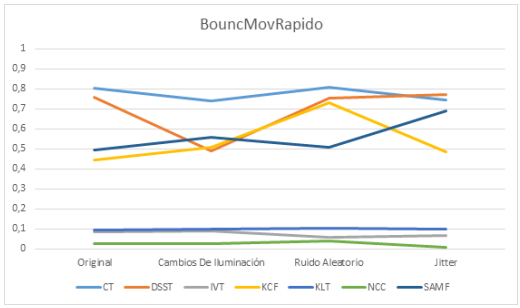
Se eligió *F1-score* como la métrica para llevar a cabo la evaluación de los algoritmos frente a las dos métricas comentadas con anterioridad debido a que *F1-score* es una evolución de ambas métricas además de englobar características de ambas. *F1-score* es una métrica cuya finalidad es calcular la exactitud con la que opera el algoritmo sobre una secuencia. Otras métricas como OTA [40] y ATA [39] también miden la exactitud con la que opera el algoritmo sobre una determinada secuencia, no obstante OTA hace uso del número de falsos negativos y de falsos positivos para hallar la exactitud, mientras que ATA se asemeja al *F1-score* puesto a que trabaja con el solapamiento entre los *bounding boxes* de *frame* a *frame*, sin embargo no realiza una medida de la precisión y del *recall*, por lo que a priori es una medida de menor exactitud.

3.3. Resultados y análisis.

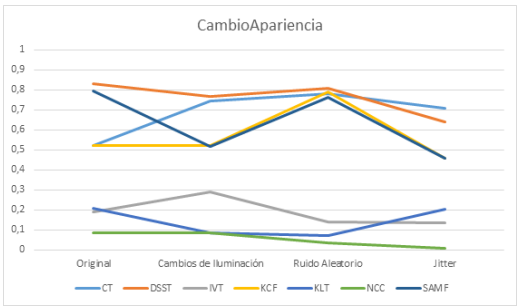
En esta sección se expondrán los resultados obtenidos mediante la métrica *F1-score* para cada algoritmo en cada secuencia. Los resultados para cada secuencia estarán organizados según el algoritmo al que pertenezcan. Se presentarán dos tipos diferentes de gráficas con información complementaria. Las primeras gráficas representarán el impacto que tiene cada efecto añadido a las secuencias (secuencias originales, secuencias con cambios de iluminación...) visto por algoritmos y para un mismo reto de *tracking* presente en las secuencias (BouncMovLento, BouncMovRapido, CambioApariencia...). El segundo tipo de gráfica tendrá como finalidad mostrar el impacto de cada reto para la media de los algoritmos en un mismo efecto generado por un UAV. En la Figura 3.1, se muestran las gráficas para cada uno de los retos de *tracking*. En estas gráficas se incluyen los resultados de cada algoritmo frente a cada efecto de UAV en dichos retos. A continuación, se analizarán en detalle estos resultados.



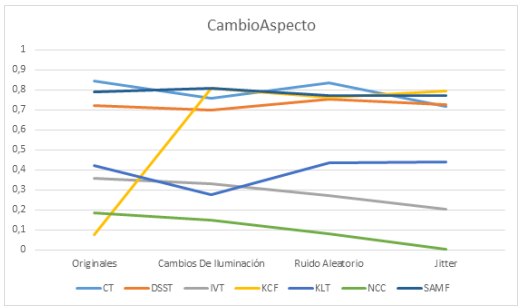
(a) Reto de movimiento de bote lento.



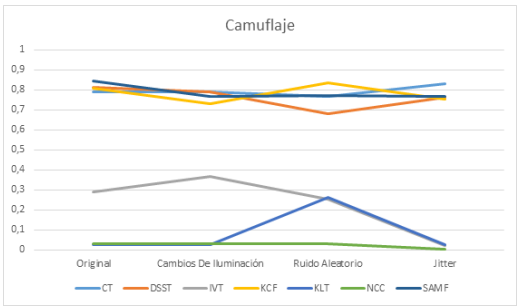
(b) Reto de movimiento de bote rápido.



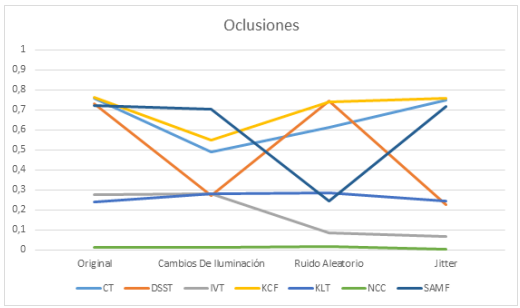
(c) Reto de cambios de apariencia.



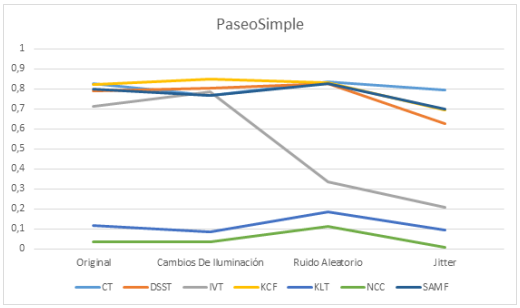
(d) Reto de cambios de aspecto.



(e) Reto de camuflaje.



(f) Reto de oclusiones.



(g) Paseo simple.

Figura 3.1: Gráficas de *F1-score* obtenidas para los retos de *tracking*.

3.3.1. Retos de tracking.

En la Figura 3.1a se aprecia que en general los resultados para el reto de movimiento de bote lento son peores para la secuencia con *jitter*, aunque existe alguna excepción como DSST en el cual mejoran los resultados con *jitter*. La secuencia original es la que obtiene los mejores resultados, mientras que para la secuencia con cambios de iluminación se aprecia cierta estabilidad, aunque en general los resultados son algo peores. Para la secuencia con ruido aleatorio los resultados son peores también aunque existe algún caso como el algoritmo KLT en el cual los resultados mejoran de forma notable.

Para el reto de movimiento de bote rápido los resultados obtenidos en la Figura 3.1b son algo confusos, debido a que la secuencia original obtiene peores puntuaciones en algunos casos que las secuencias modificadas. En la secuencia con cambios de iluminación también se obtienen resultados dispares. Mientras que para DSST y CT los resultados empeoran, para otros algoritmos como SAMF y KCF mejoran, con lo cual resulta difícil sacar conclusiones de este reto. En el caso de la secuencia con ruido aleatorio los resultados no dejan de ser sorprendentes. Cinco de los siete algoritmos mejoran los resultados de la secuencia original y en los otros dos algoritmos tampoco se aprecian diferencias muy notables. Por último la secuencia con *jitter* no iba a ser menos y también presenta unos resultados sorprendentes. En general los resultados no empeoran los obtenidos por la secuencia original, mientras que en SAMF los mejoran notablemente.

El reto de cambios de apariencia nos aporta resultados más acordes con los que a priori pensaban obtenerse, como se puede apreciar en la Figura 3.1c. En algunos casos como en CT y en KCF los resultados de la secuencia original no son los ideales en comparación con el resto de resultados, sin embargo, los resultados para el resto de efectos son los esperados. Exceptuando CT, IVT y KCF para el resto de algoritmos los resultados obtenidos para la secuencia con cambios de iluminación son peores que los de la secuencia original. Los resultados de la secuencia con ruido aleatorio son mejores que los de cambios de iluminación, pero peores que los resultados de la secuencia original, menos para los algoritmos mencionados con anterioridad. La secuencia con *jitter* obtiene los peores resultados de entre las secuencias modificadas, menos en KLT en el cual se produce una mejora notable. Los resultados también son los peores de entre todas las secuencias, si no tenemos en cuenta el *Compressive Tracking*.

Los resultados para el reto de cambios de aspecto mostrados en la Figura 3.1d son en general satisfactorios. Sorprende KCF por el resultado exageradamente malo obtenido por la secuencia original. Para la secuencia con cambios de iluminación

los resultados son peores para todos los algoritmos exceptuando KCF y SAMF. La secuencia con ruido aleatorio obtiene resultados más dispares. Mientras que para los algoritmos CT, IVT, NCC y SAMF los resultados son peores que los de la secuencia original, lo cual es de esperar, para el resto de algoritmos (DSST, KCF y KLT) estos resultados mejoran. Esto puede tener que ver con cómo están planteados los algoritmos, ya que las secuencias modificadas empleadas aquí aparentan estar suavizadas en comparación con las originales. En la secuencia con *jitter* los resultados son bastante satisfactorios. Para CT, IVT, NCC y SAMF los resultados son peores que los de la secuencia original y para el resto de algoritmos permanecen bastante estables, menos para KCF debido a la mala puntuación obtenida para la secuencia original.

Para la secuencia original los resultados obtenidos para el reto de camuflaje mostrados en la Figura 3.1e son los esperados. En la secuencia con cambios de iluminación los resultados empeoran los obtenidos para la secuencia original en todos los algoritmos menos en IVT. La de ruido aleatorio presenta unos resultados igual de satisfactorios que la secuencia con cambios de iluminación. Los resultados son peores para todos los algoritmos menos para KCF y KLT. Para la secuencia con *jitter* los resultados obtenidos también son los esperados en casi todos los algoritmos. Solo CT y KLT obtienen mejores resultados que para la secuencia original, de ellos el ciertamente relevante es CT debido a que es el que obtiene mejores resultados para todas las secuencias y todos los retos.

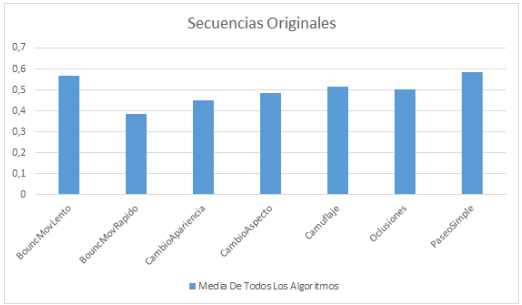
Los resultados del reto de oclusiones obtenidos por la secuencia original para todos los algoritmos son los esperados. Como se puede observar en la Figura 3.1f la secuencia con cambios de iluminación presenta unos resultados satisfactorios en cuanto a que empeoran en su mayoría los obtenidos para la secuencia original. De los algoritmos más relevantes a priori todos tienen una puntuación peor para esta secuencia, solo IVT y KLT obtienen un resultado ligeramente mejor. Con la secuencia con ruido aleatorio ocurre algo similar, solo DSST, KLT y NCC presentan resultados ligeramente mejores que los obtenidos en la secuencia original. La secuencia con *jitter* también presenta los resultados esperados. CT, DSST, IVT, KCF, NCC y SAMF obtienen peores resultados que los extraídos de la secuencia original, mientras que solo KLT alcanza un resultado ligeramente superior.

Los resultados obtenidos para el paseo simple representados en la Figura 3.1g son bastante dispares exceptuando la secuencia con *jitter* en la que en todos los algoritmos el resultado obtenido es peor que el obtenido para la secuencia original. Los resultados de la secuencia original son los esperados menos en el caso de IVT que sorprende al sacar un resultado muy bueno, tanto para la secuencia original como para la de cambios de iluminación, en comparación con los que obtuvo en todas las

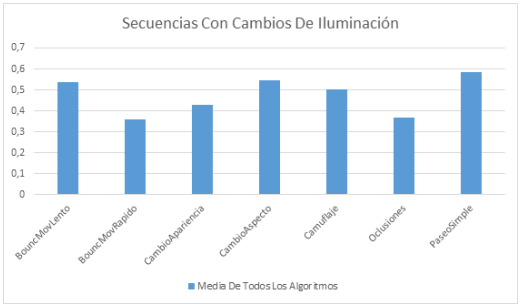
secuencias y los retos anteriores. De los algoritmos con mejores puntuaciones solo CT y SAMF obtienen un resultado peor para la secuencia con cambios de iluminación en comparación con la secuencia original. NCC y KLT también obtienen un resultado peor, pero DSST, IVT y KCF sacan resultados bastante mejores, lo cual no es lo esperado. En la secuencia con ruido aleatorio solo IVT tiene un resultado peor que en la secuencia original, el resto de algoritmos mejoran los resultados obtenidos en dicha secuencia.

3.3.2. Retos de UAVs.

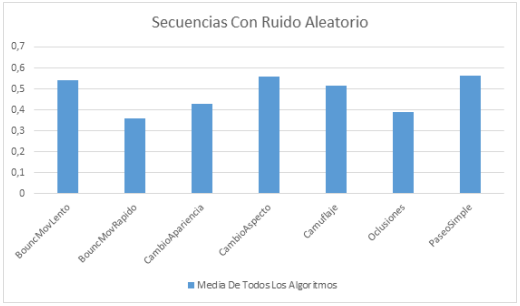
En esta subsección se van a analizar los resultados de los algoritmos de *tracking* frente a los efectos introducidos por los UAVs. En la Figura 3.2, se muestra la media de *F1-score* de todos los algoritmos por reto de *tracking* agrupados para cada efecto de UAV.



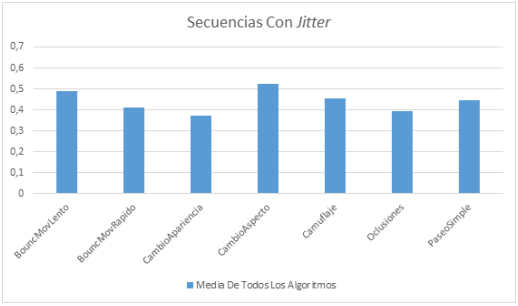
(a) Secuencias originales.



(b) Secuencias con cambios de iluminación.



(c) Secuencias con ruido aleatorio.



(d) Secuencias con *jitter*.

Figura 3.2: Media de *F1-score* de todos los algoritmos por reto para los efectos de UAV.

Como se puede extraer de los resultados obtenidos para las secuencias originales representados en la Figura 3.2a las mejores medias se dan para la secuencia de paseo simple y para el movimiento de bote lento con medias cerca de 0,6 (0,5861 y 0,5677

respectivamente). La peor media se da para el movimiento de bote rápido, la cual es la única que posee una media inferior a 0,4 (0,3866). Media general: 0,4991

La Figura3.2b, la cual representa los resultados obtenidos para las secuencias con cambios de iluminación, muestra que las medias son en general peores que las obtenidas para las secuencias originales, solo la media del reto de cambios de aspecto mejora los resultados obtenidos en el mismo reto de la secuencia original. Por lo demás los resultados son similares aunque algo peores. El paseo simple sigue siendo el que mejor media obtiene con una media cercana a 0,6 (0,5845). Le siguen el reto de cambios de aspecto y el de movimiento de bote lento con medias bastante similares (0,5467 y 0,5377 respectivamente). Las peores medias obtenidas son para los retos de movimiento de bote rápido (0,3588) y oclusiones (0,3698), ambos con medias por debajo de 0,4. Media general: 0,4756

Con las secuencias con ruido aleatorio sucede algo similar a lo que sucedía en las secuencias con cambios de iluminación. Las medias de todos los algoritmos en la Figura 3.2c empeoran en comparación con las de las secuencias originales en todos los casos menos en el del reto de cambios de aspecto, en el cual aumenta la media. La media más alta sigue siendo la del paso simple (0,5650), seguida por el reto de cambios de aspecto (0,5589) y el movimiento de bote lento (0,5423). Tampoco hay cambios en cuanto a las peores medias que son las de oclusiones (0,3900) y movimiento de bote rápido (0,3588), ambas con medias por debajo de 0,4. Media general: 0,4799

La Figura 3.2d indica que en las secuencias con *jitter* empeoran todas las medias respecto a las de las secuencias originales, menos las de los retos de cambios de aspecto y de movimiento de bote rápido, las cuales mejoran sus medias. La única media por encima de 0,5 es la del reto de cambios de aspecto que presenta una media de 0,5233. Por otra parte para este efecto las peores medias se dan en los retos de cambios de apariencia (0,3753) y de oclusiones (0,3953), con medias inferiores a 0,4. Media general: 0,442

3.4. Conclusiones.

De las gráficas representadas en la subsección 3.3.1 se pueden deducir algunas conclusiones. La primera es que, atendiendo a los resultados mostrados en dichas gráficas, tanto las secuencias que muestran el efecto de cambios de iluminación como las secuencias que muestran el efecto de ruido aleatorio obtiene en general unos resultados muy cambiantes. Si en algunos de los retos expuestos en dichas gráficas los resultados son peores que en las secuencias originales, lo cual es de esperar en un principio, en otros de estos retos los resultados mejoran. Si se observan las gráficas

de la subsección 3.3.2, las cuales muestran las medias de todos los algoritmos para cada reto en el contexto de cada efecto por separado, las conclusiones sacadas son similares. Si se comparan una a una las medias obtenidas para cada uno de los retos de las secuencias originales y las secuencias con cambios de iluminación, se aprecia un empeoramiento de las medias para las secuencias con cambios de iluminación, sin embargo, este empeoramiento solo es notable en la secuencia con oclusiones, mientras que en la secuencia con cambios de aspecto la mejora también es notable. Aunque las medias son en general peores que las obtenidas para las secuencias originales, esta caída de las medias no es tan grande como para preocuparse por corregir el efecto presente en estas secuencias. Si en las secuencias con cambios de iluminación las diferencias entre las medias de estas secuencias con las de las originales no son grandes, con las secuencias con ruido aleatorio ocurre algo similar, sin embargo, en este caso las diferencias son incluso más pequeñas.

Si se pone la atención en las secuencias con *jitter* para ambos tipos de gráfica se aprecia una caída clara de los resultados. Para las gráficas de la subsección 3.3.1, si bien es cierto que existen algunas excepciones como se ha explicado para cada una de las gráficas en dicha subsección, en general la tendencia de la curva es descendente cuando se llega a la secuencia con *jitter*. En base a estos análisis, se llega a la conclusión de que las secuencias con *jitter* son las que obtienen los peores resultados de entre las secuencias modificadas y sin modificar. En la Figura 3.2d dedicada exclusivamente al efecto de *jitter* las diferencias se hacen aún más visibles. Las medias obtenidas son bastante peores que las obtenidas en la secuencia original excepto en los retos de movimiento de bote rápido y de cambios de aspecto. Pero además de ser peores que las medias de las secuencias originales, también son las peores prácticamente en la totalidad de los retos de entre las secuencias modificadas, siendo estas diferencias bastante notables en algunos de los retos presentes en la gráfica.

Además de los efectos simulados y evaluados en este *toy example* existe un efecto presente en algunas de las secuencias grabadas y que es inherente a las cámaras de grabación: el *blurring*. Este efecto es más palpable a medida que se reduce la calidad de la cámara con la que se está grabando. Para grabar las secuencias de este *toy example* se hizo uso de una cámara de móvil, por lo que el *blurring* aparece durante algunos *frames* en algunas de las secuencias grabadas.

Teniendo en cuenta el *jitter* y el *blurring* la idea consiste en realizar una corrección de las secuencias con *jitter* que tengan en cuenta el efecto de *blurring* para posteriormente evaluarlos en comparación con las secuencias modificadas con *jitter* y ver si se han conseguido mejorar dichos resultados.

Capítulo 4

Propuesta de mejora.

4.1. Visión general del sistema.

La Figura 4.1 representa los cinco componentes más habituales en los algoritmos de *tracking*, así como el flujo de información que sigue el algoritmo. Estos componentes están habitualmente presentes en los *trackers* independientemente de las técnicas concretas empleadas para estimar la posición del objetivo.

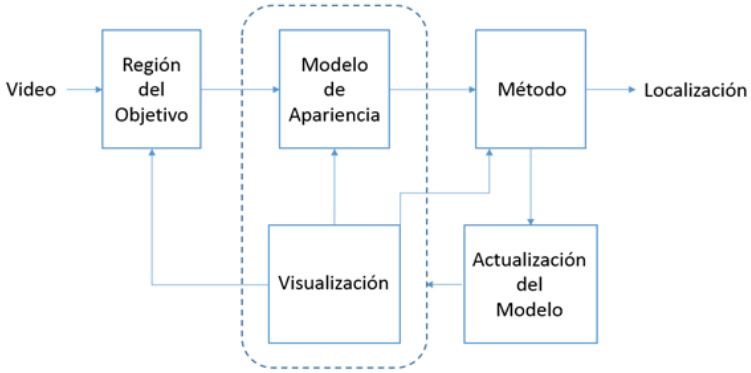


Figura 4.1: Diagrama de bloques del sistema.

A la entrada del diagrama tenemos la secuencia de vídeo. El primer componente del diagrama consiste en obtener la región en la que se encuentra el objeto al que se pretende seguir. El comportamiento del algoritmo en el primer cuadro variará según el algoritmo. En algunos casos el propio algoritmo lo estimará, mientras que en otros habrá que proveerle la información manualmente. Destacar que los algoritmos con los que se ha trabajado necesitan que se les proporcione la posición inicial del objeto a seguir.

Una vez se dispone de la ubicación en el primer cuadro de la secuencia de vídeo se puede comenzar a representar la imagen en conjunto con la localización del elemento de interés, empleando el modo de salida escogido, un *bounding box* es la técnica más habitual. De nuevo todos los algoritmos escogidos compartirán una misma aproximación a la hora de dar sus resultados, *bounding box*.

Habitualmente haciendo uso de la información disponible, se pasa al algoritmo propiamente dicho, el cual ya es característico de cada aproximación. Dentro del método se suelen realizar diversas medidas, como pueden ser comparaciones entre imágenes o extracción de características, previamente obtenidas en el módulo del modelo de apariencia, para obtener como resultado la localización del objetivo en la siguiente imagen. Esta información de localización es lo que habitualmente se almacena para la posterior evaluación del algoritmo frente la información de *ground-truth* en caso de que ésta estuviera disponible.

Por último, y de nuevo dependiendo del algoritmo, en ocasiones se procede a actualizar las medidas y/o modelos de características.

Todos estos pasos explicados con anterioridad se repetirán para todos los *frames* de la secuencia de vídeo, obteniendo como resultado final del *tracking* de toda la secuencia la localización estimada del objetivo en cada *frame* de dicha secuencia.

4.2. Mejoras introducidas

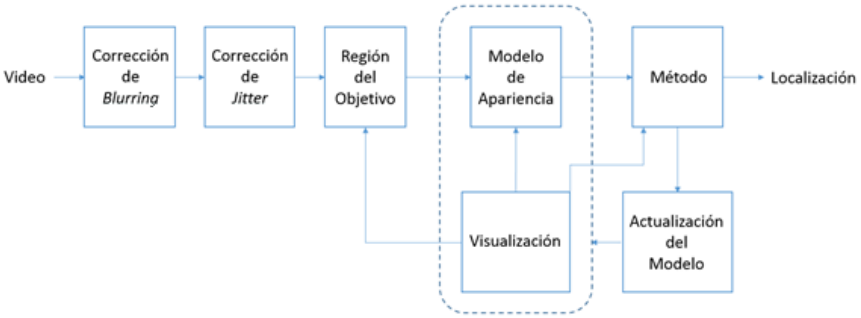


Figura 4.2: Diagrama de bloques del sistema con las correcciones añadidas.

A continuación, se va a pasar a exponer las modificaciones introducidas con el objetivo de mejorar el comportamiento de los algoritmos de *tracking* evaluados a funcionar en situaciones como las expuestas en la evaluación del *toy-example*. Ambas mejoras, enfocadas al efecto de *blurring* y *jitter*, tendrán un carácter de pre-procesado, y su

objetivo se presenta en cada una de las correspondientes subsecciones, si bien su motivación viene de la evaluación realizada anteriormente. En la Figura 4.2, se muestra un diagrama de bloques del sistema completo con las modificaciones introuducidas.

4.2.1. Mejora frente al efecto de blurring.

Como se explicó anteriormente, el *blurring* aparece en varias de las secuencias grabadas para el *toy example* y resulta determinante afrontar este efecto para trabajar con imágenes de mejor calidad en la próxima fase.

Según el diagrama de bloques de la Figura 4.2 a la entrada de este bloque se tiene el vídeo, el cual puede disponer de imágenes difusas, con poco contraste y con los bordes poco definidos. El objetivo del bloque de corrección de *blurring* es conseguir a la salida de dicho bloque una imagen con más contraste para que más adelante sea más sencillo detectar puntos de interés dentro de la propia imagen.

Para ello se hace uso de una función, la cual aplicada a una imagen poco contrastada devuelva la misma imagen con características propias, como por ejemplo los bordes, más contrastados. El método en el que se basa esta función es *unsharp masking*. Este método consiste en restar pixel por pixel la imagen original con una versión suavizada de la misma imagen para obtener una imagen auxiliar, la cual finalmente será sumada a la imagen original dando lugar de este modo a una versión con alto contraste de la propia imagen. Otro método que lleva a una solución similar es sumar la imagen original a una versión escalada de la misma imagen a la que se le ha aplicado un filtro paso alto [41].

4.2.2. Mejora frente al efecto de jitter.

Tal y como se dedujo de la sección 3.4 sería positivo tratar de corregir el efecto que pueda tener el *jitter* en una secuencia debido a que la presencia de dicho efecto afecta de manera notable a los resultados obtenidos por el *tracker* a la hora de seguir a un objeto. En esta subsección se procederá a explicar las mejoras realizadas para corregir el efecto que produce el *jitter*.

Primeramente es importante explicar a qué nivel se han llevado a cabo las mejoras. Dentro de la Figura 4.2 las mejoras establecidas se encuentran antes del componente de la obtención de la región en la que se encuentra el objeto a seguir, exactamente entre la entrada, que es el vídeo después de aplicarle la mejora del efecto de *blurring*, y el primer componente del diagrama. Es decir, el efecto de *jitter* se corregirá directamente sobre la secuencia completa que contiene dicho efecto. El objetivo a la salida del bloque de corrección de *jitter* es obtener una secuencia de vídeo en la que el efecto de *jitter*

no sea apreciable y que por lo tanto el movimiento del objetivo dentro de la secuencia sea más sencillo de estimar.

Para poder llevar a cabo la reconstrucción de la secuencia se tienen que realizar varias etapas, las cuales se han asociado a diferentes funciones específicas. La primera de las etapas es la de detección de esquinas en la imagen. A la entrada de esta etapa se dispondrá de dos imágenes consecutivas modificadas para tener un alto contraste y el objetivo consiste en obtener puntos de interés dentro de ambas imágenes. A la función que realiza esta etapa se le pasan las imágenes modificadas junto a un parámetro que determina la intensidad mínima que puede haber entre la esquina a detectar y la región que lo rodea, de modo que al aumentar este parámetro se detectan menos esquinas. A la salida de dicha función se obtiene un objeto para cada imagen en el cual se encuentran las localizaciones en nivel de grises y 2-D de las esquinas detectadas. Para detectar las esquinas la función hace uso del detector de características FAST (*Features from Accelerated Segment Test*) [42], el cual compara los 16 píxeles contiguos al pixel que se está testando para detectar diferentes características. Si doce de los píxeles contiguos poseen valores por encima o por debajo de cierto umbral se detecta una característica en ese pixel. La ventaja de este detector es que el testeo para saber si en cierto pixel se detecta una característica no es necesario llevarlo a cabo en todos los píxeles, debido a que si ya se sabe cómo son los píxeles alrededor del pixel testeado con anterioridad algunos ya se pueden desechar al ser obvio que no se va a obtener ninguna característica en esa posición. Después de detectar las esquinas y obtener sus localizaciones se desecharon las esquinas presentes dentro del *ground-truth* de la secuencia, debido a que el movimiento del objeto a seguir afectaba de gran manera a la secuencia reconstruida.

Una vez se tienen los puntos de interés de la imagen el siguiente paso consistirá en extraer los *feature vectors*, también conocidos como descriptores, para cada uno de estos puntos, información que será clave para la etapa de búsqueda de puntos comunes entre imágenes consecutivas. A la entrada de esta etapa se tienen la imagen y la localización de los puntos de interés calculados para esa imagen. Esto se hace para dos imágenes consecutivas al igual que en la etapa de detección de puntos de interés. La función devolverá los descriptores y sus localizaciones correspondientes. Como método de extracción de descriptores se ha empleado el método FREAK (*Fast Retina Keypoint*) [43]. El método FREAK emplea un patrón de muestreo similar al funcionamiento de la retina. La retina está compuesta por células ganglionares, las cuales reciben información de los fotorreceptores. En el centro de la retina, la foveola, la densidad de células ganglionares es alta, reduciéndose con un ritmo exponencial al alejarse del centro. Los creadores de FREAK adoptaron esta idea e hicieron uso de una

rejilla circular de mayor densidad de puntos en el centro, reduciéndose la densidad de puntos exponencialmente al alejarse del centro para obtener los descriptores de cada punto de interés. FREAK devuelve como descriptor la orientación del punto de interés en radianes.

La siguiente etapa hace uso de una función que tiene a la entrada las características extraídas para las dos imágenes consecutivas en la anterior etapa. A la salida se obtendrá una matriz del tamaño P-por-2 siendo P el número de pares de características coincidentes con los índices de las características que dicha función ha considerado que son coincidentes en ambas imágenes. Para encontrar los puntos coincidentes en la imagen realiza una búsqueda paralela de los vecinos más cercanos de dicho punto. El sistema en el que se buscan los vecinos más cercanos consiste en un entorno de árboles jerárquicos en los que se agrupan todos los puntos de interés de la imagen. Para construir estos árboles se crea un nodo hoja (nodo final) en caso de que el número de puntos de interés D sea menor que el tamaño máximo de nodos hoja SL [44]. Posteriormente se eligen K puntos aleatorios dentro de la totalidad de los puntos de interés que serán los centros de las agrupaciones de puntos, asignando a cada agrupación los puntos más cercanos al centro de una agrupación en comparación con las otras. En cada una de estas agrupaciones se realizarán de nuevo los mismos pasos hasta que el número de puntos dentro de cada agrupación alcance el tamaño máximo de nodos hoja. La búsqueda de los vecinos más cercanos comienza explorando el nodo más cercano al punto del que se desea obtener su coincidente. Los nodos que no han sido explorados se añaden a una cola de prioridad que será empleada más adelante en la búsqueda. Una vez se alcanza el nodo hoja se exploran todos los puntos contenidos en dicha agrupación. Una vez se ha finalizado la búsqueda en cada uno de los árboles se volverá a llevar a cabo el mismo proceso para todos los nodos almacenados en la cola de prioridad. Esta búsqueda finaliza una vez se ha llegado a un número determinado de puntos examinados, información que nos proporciona un parámetro de la función, el cual está directamente relacionado con la precisión de la búsqueda, puesto a que cuanto mayor es el número de puntos explorados mayor es también la precisión de los vecinos encontrados.

Una vez tenemos las combinaciones de puntos que coinciden en ambas imágenes consecutivas se necesitará saber cuáles de estos puntos son realmente coincidentes para poder reconstruir la imagen y cuáles son erróneos y pueden llevar a una falsa interpretación de cómo ha cambiado la imagen de una imagen a la siguiente. Para ello se hace uso de una función, la cual a la entrada tiene los puntos de ambas imágenes ordenados según su coincidencia entre ellos, así como un parámetro que establece el número mínimo de puntos coincidentes para establecer la transformación. En esta

ocasión se escogió como valor del parámetro un número mínimo de tres pares de puntos coincidentes. La función hace uso de la geometría proyectiva y de la transformación proyectiva 2-D [45] para determinar qué pares de puntos son los correctos. Finalmente devuelve los pares de puntos coincidentes correctos en dos vectores, cada uno para su imagen, además de una matriz que representa la transformación geométrica y que contiene el mapeo entre los puntos presentes a la entrada de la función.

Antes de aplicar la corrección solo queda la etapa en la que se calcula la homografía. La homografía es la transformación proyectiva que establece la correspondencia entre dos planos. Rotación, traslación y escala son los parámetros empleados para determinar dicha transformación. Los tres parámetros se extraen de la transformación geométrica obtenida como resultado de la anterior función, calculando para cada par de imágenes una nueva homografía.

La penúltima etapa por la que hay que pasar antes de tener una de las imágenes de la secuencia corregida es la de aplicado de la transformación. En esta etapa se pretende aplicar todos los datos obtenidos con anterioridad para adquirir la imagen corregida. A esta etapa se le entregan a la entrada la imagen que se quiere transformar junto a la transformación geométrica calculada con anterioridad, además de otros dos argumentos que determinan el tamaño y la localización de la imagen obtenidas a la salida de la función en el sistema global de coordenadas. La función se encarga de transformar la imagen a la entrada mediante la transformación geométrica, dando como resultado una imagen corregida a la salida.

La imagen obtenida a la salida de la anterior etapa contiene en muchos casos zonas negras en los extremos de la imagen presentes debido al corregido de la misma. Esto puede ser perjudicial para la evaluación que se quiere llevar a cabo debido a que uno de los retos a cubrir es el de las oclusiones, sin embargo, en caso de no corregir los extremos prácticamente todas las secuencias contarían con oclusiones totales y/o parciales del objeto a seguir y por lo tanto la evaluación no sería la correcta. El objetivo por lo tanto consiste en reconstruir estas zonas con el fin de evitar que el algoritmo de búsqueda tenga que lidiar con oclusiones cada vez que el objeto a seguir se mueva por las zonas cercanas a los extremos de la imagen. Para ello se ha optado por usar un sistema sencillo de replicado de pixels, el cual consiste en copiar los pixels anteriores a las zonas negras en esos huecos. Este proceso se realiza primero en las partes inferior y superior de la imagen y luego en los extremos laterales, dando como resultado una imagen sin grandes regiones negras y con un relleno razonablemente coherente en la mayoría de las secuencias, pudiendo así evaluar dichas secuencias de forma correcta.

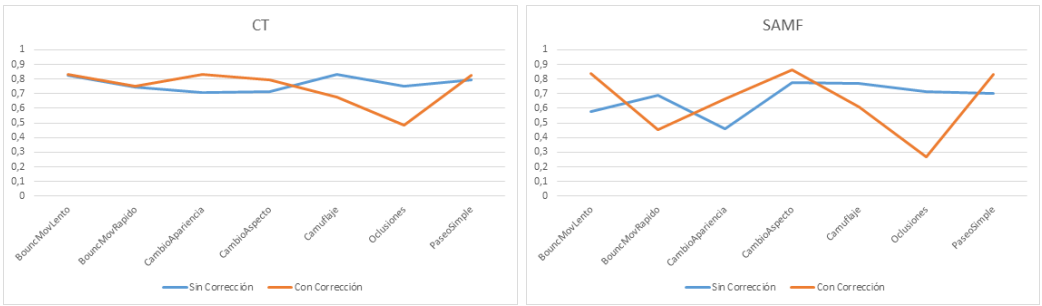
4.3. Evaluación comparativa en Toy example.

En esta sección se han aplicado las correcciones sobre las secuencias con el objetivo de evaluarlas. Se pretende demostrar que al aplicar la corrección del *jitter* sobre cada secuencia y por lo tanto para cada reto los resultados mejorarán en comparación con los obtenidos para las secuencias sin corregir.

A continuación se mostrarán los resultados obtenidos para las secuencias en las que se ha corregido el *jitter*.

Para comparar los resultados obtenidos por la secuencia con *jitter* para cada reto de *tracking* con el resto de secuencias se hará uso de dos tipos de gráficas. Estas gráficas serán similares a las gráficas de la subsección 3.3.2 empleadas en el *toy example*, mostrando en las primeras los resultados obtenidos por las secuencias con *jitter* corregido en comparación con los resultados de las secuencias sin corregir en cada uno de los retos para los algoritmos CT y SAMF, mientras que en el segundo tipo de gráfica se mostrará una comparación entre la media de los resultados de todos los algoritmos para todos los retos en las secuencias con *jitter* corregido y sin corregir. Se ha decidido mostrar, en el primer tipo de gráfica, los resultados de CT y SAMF, debido a que se trata de los algoritmos con los resultados más estables y más altos según se ha podido observar en los resultados del *toy example*.

En la Figura 4.3a se puede apreciar que el algoritmo CT arroja los mejores resultados para las secuencias con corrección de *jitter*. En cinco de los siete retos sobre los que se ha evaluado este algoritmo los resultados han sido mejores empleando las secuencias con *jitter* corregido en lugar de las secuencias sin corregir. Solo en los retos de camuflaje y de oclusiones los resultados han sido peores. Los retos para los que se han obtenido mejores resultados en las secuencias corregidas son el de movimiento de bote lento y el de cambios de apariencia, reto para el cual la diferencia con las



(a) Algoritmo CT frente a los retos de *tracking*. (b) Algoritmo SAMF frente a los retos de *tracking*.

Figura 4.3: Resultados de *F1-score* obtenidos por los algoritmos.

secuencias sin corregir es la mayor de entre los cinco retos con resultados superiores para las secuencias corregidas. La diferencia más abultada se da en el reto de las oclusiones, esta vez con un resultado desfavorable para las secuencias corregidas.

Los resultados obtenidos para el algoritmo SAMF y mostrados en la Figura 4.3b son algo dispares. Se han adquirido mejores resultados en las secuencias con corrección en cuatro de los siete retos, siendo los retos de movimiento de bote rápido, camuflaje y oclusiones los que obtienen resultados por debajo de los de las secuencias sin corregir. La mayor diferencia a favor de las secuencias con corrección se ha producido en el movimiento de bote lento, mientras que la mayor diferencia a favor de las secuencias sin corregir ha vuelto a aparecer en el reto de las oclusiones.

En la gráfica de la Figura 4.4 se puede observar que los resultados de las secuencias corregidas son mejores que los de las secuencias sin corrección. En cuatro de los siete retos evaluados la media lograda por todos los algoritmos para las secuencias corregidas es mayor que la media de las secuencias sin corrección. Esta mejora de las medias es bastante notable en los retos de movimiento de bote lento y de paseo simple, mientras que en los retos de cambios de apariencia y de aspecto se aprecia menos, aunque también sean mejores. En el reto de movimiento de bote rápido es donde más baja la media de las secuencias corregidas, siendo menos notable esta caída en los retos de camuflaje y de oclusiones. Otro dato que confirma la mejora general de los resultados para las secuencias corregidas es que si se hace la media de las diferencias de todos los retos el resultado es 0,0187, es decir que si se corrigen las secuencias con *jitter* haciendo uso del procedimiento explicado con anterioridad, los resultados mejoran un 0,6 % de media.

Los resultados obtenidos para cada reto de *tracking* y efecto de UAV de las secuencias del *toy example*, así como los resultados de las secuencias corregidas ordenados según el algoritmo de *tracking* con el que han sido evaluados se encuentran en las tablas del Anexo B.

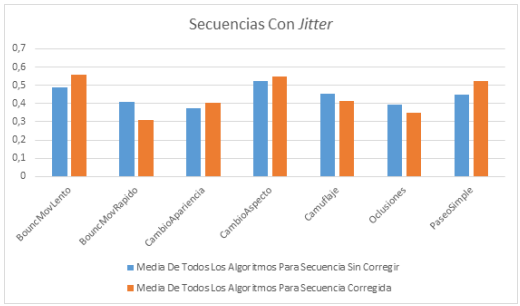


Figura 4.4: Comparativa de la media de *F1-score* de todos los algoritmos entre las secuencias corregidas y las originales para cada reto.

4.4. Conclusiones

Teniendo en cuenta los resultados mostrados en la sección 4.3 se puede decir que corregir las secuencias ha servido para mejorar el rendimiento que tenían los algoritmos en las secuencias con *jitter*. Si bien es cierto, esta mejora no se ha podido apreciar en todos los retos evaluados, el global de los resultados apunta hacia una mejora de los mismos.

La mejora de los resultados se ha dado en las secuencias en las que el movimiento no era excesivamente rápido e impredecible ni existía ningún elemento potencialmente distractor para el *tracker* como pueda ser la presencia de una persona ocluyendo parte del objetivo. Por lo tanto se puede decir que la mejora aparece sobre todo cuando el movimiento es fácil de predecir, como en las secuencias con el reto de movimiento de bote lento y de paseo simple, o cuando el objetivo es distinguible, a pesar de que pueda haber cambios en la apariencia o aspecto del objetivo.

En el caso de los resultados obtenidos por el algoritmo SAMF, éstos pueden verse muy afectados por cómo está construido el algoritmo. Como se menciona en el Anexo A, SAMF es un algoritmo que se ve fuertemente afectado tanto por los cambios de escala como por las oclusiones. De este modo se puede explicar por qué en algunos de los retos los resultados son peores para las secuencias corregidas y es debido a que dichas secuencias contienen cierto componente de cambios de escala. A su vez el reto de las oclusiones contiene en su secuencia corregida tanto oclusiones como cambios de escala, por lo que el resultado obtenido no es sorprendente.

Otro elemento que puede explicar el mal rendimiento que tienen algunos algoritmos para el reto de las oclusiones es la técnica de *warping* [46]. Esta técnica busca combatir los problemas de los algoritmos con los cambios de escala, pero a costa de empeorar en retos como el de las oclusiones. Consiste en buscar dentro de un cierto rango de píxeles vecinos la máscara que mejor se acople a la máscara anterior dando lugar, de este modo, a posibles confusiones en oclusiones con elementos de cierto parecido.

Los resultados visibles en la Figura 4.4 muestran que el reto del camuflaje también sufre cierto empeoramiento respecto a las secuencias sin corregir. Esto puede ser debido a como se corrigió la secuencia en sí. Al corregir la secuencia se pone especial énfasis en que ninguno de los puntos de interés detectados se encuentre encima del objetivo, puesto a que el movimiento del mismo simularía un movimiento de la cámara que no es realista. En el reto de camuflaje sucede que alguno de los puntos de

interés detectados se encuentra encima del objetivo debido a que no se es capaz de distinguir entre objetivo y fondo, dando lugar a una corrección de la secuencia errónea y obteniendo como resultado un empeoramiento en la detección de los algoritmos.

El otro reto que muestra empeoramiento en sus resultados en comparación con los obtenidos para las secuencias sin corrección es el de movimiento de bote rápido. En este caso la explicación de por qué los resultados han empeorado es más intuitiva. Al tener la secuencia sin corregir *jitter* en algunas ocasiones el cambio de dirección en el movimiento del objetivo para simular el movimiento de bote coincide con que el *jitter* en ese cuadro adoptó una dirección contraria a la del movimiento del objetivo, suavizando de este modo el movimiento repentino del objetivo y favoreciendo el seguimiento por parte de los algoritmos.

Capítulo 5

Evaluación y resultados.

Una vez se han evaluado las secuencias para cada uno de los retos y se ha corregido el *blurring* y el *jitter* para las secuencias grabadas en el *toy example* el siguiente paso consiste en realizar los mismos pasos con secuencias reales grabadas desde un UAV para averiguar si las mejoras introducidas conllevan un incremento en los resultados obtenidos y por lo tanto pueden ser aplicables. El objetivo por lo tanto consiste en aplicar las correcciones de *blurring* y de *jitter* que mejoraban los resultados obtenidos por las secuencias con *jitter* en el *toy example* a las secuencias grabadas desde un UAV y ver si los resultados obtenidos mejoran los de las secuencias sin modificar.

5.1. Framework de evaluación.

5.1.1. Dataset.

Las grabaciones llevadas a cabo han sido captadas desde una cámara dispuesta en un UAV. Al ser grabadas desde un dron los retos debidos al UAV ya están presentes en las secuencias captadas y por lo tanto no es necesario simularlas como se hizo en el *toy example*. Otra diferencia con las secuencias del *toy example* es que también hay movimiento de la cámara debido al movimiento del UAV, lo cual puede complicar las cosas a la hora de corregir el *jitter* de las secuencias.

Se han grabado siete secuencias diferentes desde una altura similar a la de la cabeza del objetivo y una distancia de unos seis metros. Se decidió grabar a esa altura y distancia debido a que si el UAV se encuentra a una altura y distancia lejanas, cualquier pequeño movimiento en el UAV significa un gran cambio en la posición del objetivo, pudiendo afectar de gran manera a la estabilización y corrección de la imagen. De entre las secuencias presentes en el *toy example* la única que no está representada en estas secuencias es la de paseo simple, sin embargo el resto de retos

Retos	Nº de frames
Bounc_Mov_Lento	141
Bounc_Mov_Rapido	148
Cambio_Apariencia	185
Cambio_Aspecto	319
Camuflaje_Soft_Color	168
Camuflaje_Strong_Color	229
Oclusiones	129

Tabla 5.1: Secuencias reales y número de *frames* de cada una.

son los mismos: movimiento de bote lento, movimiento de bote rápido, cambios de apariencia, cambios de aspecto, camuflaje y oclusiones. En esta ocasión se ha decidido grabar dos secuencias diferentes para el reto de camuflaje. Una de ellas llevando el objetivo prendas oscuras y por lo tanto muy difíciles de distinguir en las sombras y la otra con prendas de un color más llamativo siendo a priori algo más fáciles de distinguir en las zonas sombrías. Cada una de estas secuencias se ha grabado de forma independiente mostrando un solo reto por cada secuencia.

En la Tabla 5.1 se muestra el número de *frames* para cada secuencia.

El número total de *frames* para todas las secuencias es 1'319. Todas las secuencias han sido grabadas en formato AVI con un ratio de 25 cuadros por segundo y una resolución de 960x540.

La secuencia más corta de todas es la del reto de oclusiones, la cual, a diferencia de como se hizo en el *toy example,* se ha realizado empleando un objeto fijo, en este caso un árbol, para ocluirse en lugar de una persona. El objetivo realizó movimientos de lado a lado por la parte de atrás del árbol, ocluyéndose parcialmente, continuando posteriormente la marcha. La secuencia más larga es la que simula el reto de cambios de aspecto. En esta secuencia el objetivo va andando mientras realiza diferentes movimientos con los brazos pasando a agacharse para luego levantarse de nuevo en dos ocasiones. La escena grabada para el reto de camuflaje con prendas oscuras es igual que la grabada con prendas de colores fuertes. Ambas consisten en una persona andando desde una zona iluminada hacia una zona con sombra generando de esta manera el camuflaje del objetivo. Lo único que cambia entre una secuencia y la otra son las prendas que usa el objetivo. En la secuencia de movimiento de bote lento el objetivo realiza varios cambios de dirección imprevisibles para el algoritmo a un ritmo constante y normal. Esto cambia para la secuencia de movimiento de bote rápido en la que el objetivo realiza del mismo modo cambios de dirección, pero esta vez unidos a cambios de ritmo en el paso del objetivo. La secuencia que simula el reto de cam-

bios de apariencia es la más similar a la grabada en el *toy example*. Al igual que en la secuencia grabada con cámara fija el objetivo pasea hasta el centro de la escena procediendo a quitarse la chaqueta para luego continuar la marcha.

Una vez se tienen las secuencias grabadas se ha procedido a pre-procesarlas para anotarlas con el fin de usarlas para la evaluación de los diferentes algoritmos de *tracking*. Todas las secuencias han sido anotadas del mismo modo que se hizo para las secuencias del *toy example* empleando la herramienta de anotación *VIA*. La anotación se ha realizado *frame* por *frame* atendiendo al tamaño y movimiento del objetivo, por lo que el *bounding box* anotado tiene un tamaño variable al igual que en el *toy example*. Una vez se ha anotado cada secuencia se ha extraído el archivo con el *ground-truth* para cada una de estas secuencias procesándolo con el fin de dejarlo en el formato deseado.

5.1.2. Métricas.

Al igual que se hizo en el *toy example* la métrica elegida para evaluar tanto las secuencias grabadas como las corregidas será el *F1-score*.

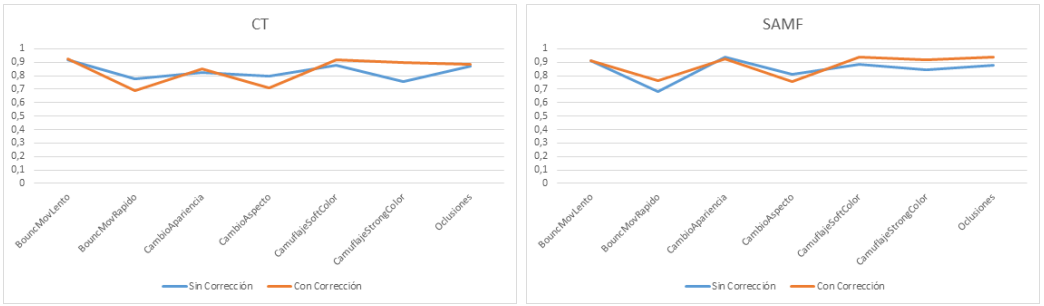
5.2. Resultados y análisis.

A continuación se mostrarán los resultados obtenidos tanto para las secuencias grabadas como para las secuencias corregidas realizando una comparación entre ambas con el fin de averiguar si al corregir las secuencias se consiguen mejorar los resultados, y en caso de haber mejora mostrar el porcentaje en el que se han mejorado los resultados. Al igual que en la sección 4.3 en la cual se evaluaban las secuencias corregidas en comparación con las grabadas en el *toy example*, los algoritmos empleados para llevar a cabo la comparación en este caso serán CT y SAMF, puesto a que se trata de los algoritmos que mejores resultados obtienen en general.

Las gráficas a mostrar serán similares a las mostradas en la sección 4.3. Las primeras dos gráficas mostrarán una comparación entre los resultados obtenidos por las secuencias grabadas y las secuencias corregidas para cada uno de los retos a evaluar, tanto para CT como para SAMF, mientras que en la última gráfica se van a comparar las medias obtenidas por todos los algoritmos para cada reto. De este modo se podrá observar tanto de forma individualizada como de forma global el efecto que tiene corregir las secuencias.

En la comparación de los resultados obtenidos para ambos tipos de secuencia por el algoritmo CT en la Figura5.1a salen ganando las secuencias corregidas en cinco de los siete retos evaluados, por lo que atendiendo a estos resultados CT funciona mejor

en las secuencias en las que se ha corregido el *jitter*. Los únicos dos retos en los que los resultados empeoran para las secuencias corregidas son el de movimiento de bote rápido y el de cambios de aspecto, siendo la diferencia de 0,1 aproximadamente. En el resto de retos los resultados mejoran, dándose la mejora más notable en el reto de camuflaje con colores llamativos con una mejora de prácticamente 0,15.



(a) Algoritmo CT para cada uno de los retos de *tracking*.

(b) Algoritmo SAMF para cada uno de los retos de *tracking*.

Figura 5.1: Resultados de *F1-score* obtenidos por los algoritmos.

Los resultados obtenidos por el algoritmo SAMF en la Figura5.1b muestran una mejora de los resultados para las secuencias corregidas en cuatro de los siete retos para las secuencias grabadas. Los retos para los que los resultados de las secuencias empeoran son los de movimiento de bote lento, cambios de apariencia y cambios de aspecto, mientras que para el resto de retos evaluados los resultados mejoran. En este algoritmo las diferencias son más grandes para los retos en los que los resultados han mejorado que en los que han empeorado, dándose la diferencia más amplia en el reto de movimiento de bote rápido con un valor de aproximadamente 0,08. La diferencia más amplia entre los retos en los que empeoran los resultados para las secuencias corregidas se da en el reto de cambios de aspecto con un valor de 0,06.

En la Figura 5.2 se puede apreciar que los resultados de las secuencias corregidas son mejores que los de las secuencias sin corregir, si se hace la media de los resultados obtenidos por todos los algoritmos para cada uno de los retos. La media obtenida por las secuencias corregidas es superior en cuatro de los siete retos evaluados. La mejora más abultada se da para el reto de las oclusiones, mientras que para los dos retos de camuflaje también es bastante notable. El otro reto para el cual la media es mejor es el de movimiento de bote lento, aunque en este caso la diferencia es pequeña. Al igual que sucedía en el *toy example* el reto de movimiento de bote rápido es en el que más empeora la media de los resultados, siendo los retos de cambios de apariencia y de aspecto los otros dos retos en los que la media empeora. Un dato que confirma la

mejora de los resultados al corregir el *jitter* de las imágenes es que la media de las diferencias entre las secuencias con corrección y sin corrección para cada reto es 0,0686 a favor de las secuencias corregidas, lo cual indica que llevar a cabo una corrección del *jitter* del modo que se ha explicado sobre secuencias grabadas mediante un UAV mejoran los resultados un 1,8325 % de media.

Los resultados obtenidos para cada reto de *tracking* por cada algoritmo en las secuencias reales corregidas y sin corregir se encuentran en las tablas del Anexo C.

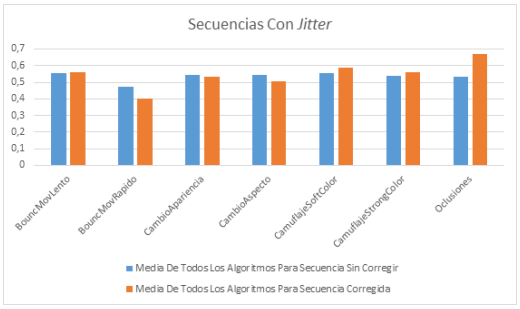


Figura 5.2: Comparativa de la media de *F1-score* de todos los algoritmos entre las secuencias corregidas y las originales para cada reto.

5.3. Conclusiones

Para observar si las correcciones introducidas en el *toy example* se podían aplicar de forma real, se llevaron a cabo varias grabaciones realistas mediante un UAV con el objetivo de evaluar las correcciones. Antes de grabar las secuencias se pensó en realizarlas de un modo que favoreciese el seguimiento del objetivo, como emplear en muchos de los retos colores diferenciados con el fondo o evitar la presencia de otra persona en la grabación, lo cual ha hecho que los resultados globales tanto para las secuencias sin corregir como para las secuencias corregidas sean en general mejores que los obtenidos en el *toy example*. Además de esto se grabaron todas las secuencias desde una altura similar a la del objetivo y una distancia corta debido a que de este modo se apreciaría más la estabilización y corrección de la imagen, puesto a que si se grabase desde mucha distancia y altura, los efectos del *jittter* introducidos por el UAV serían prácticamente inapreciables en relación a los cambios bruscos de posición que se sufren a esas alturas, por lo que la corrección no tendría mucho sentido.

Como se ha podido observar en la sección 5.2 los resultados vistos desde un punto de vista global han mejorado para las secuencias corregidas. Esta mejora no es apre-

ciable en todos los retos evaluados, sin embargo sí en la mayoría. Además de esta mejora vista de reto en reto, también existe una mejora en la media de las diferencias obtenidas por todos los algoritmos para cada reto, siendo esta mejora de casi un 2 %, por lo que también en este ámbito los resultados obtenidos en las secuencias reales superan los obtenidos en el *toy example*.

A diferencia de como sucedía en el *toy example*, los retos de cambios de apariencia y de aspecto sufren un empeoramiento de los resultados globales, atendiendo a la Figura 5.2, mientras que los retos de camuflaje y de oclusiones mejoran de forma notable en el caso del camuflaje y abultada en el caso de las oclusiones.

Los únicos retos que obtienen unos resultados similares a los obtenidos en el *toy example* son los de movimiento de bote. En el caso del movimiento de bote rápido los resultados empeoran de forma notable, tanto para los algoritmos SAMF y CT, como para la media de todos los algoritmos. De hecho dicha media es la peor de todas las obtenidas y supone la mayor diferencia en contra de las secuencias corregidas. Como ya se explicó cuando se evaluaron las correcciones aplicadas al *toy example* este empeoramiento se debe a cómo está grabada la secuencia en sí. El *jitter* unido a cambios repentinos en el movimiento del objetivo puede llevar a un suavizado del movimiento del mismo al producirse dicho movimiento en contra del movimiento del UAV. En el caso del movimiento de bote lento los resultados mejoran tanto para los dos algoritmos mostrados de forma individual como para el global de los algoritmos, aunque esta mejora sea pequeña.

Con el reto de cambios de apariencia sucede algo similar a lo que sucede con el movimiento de bote lento, y es que aunque se produce un empeoramiento de los resultados para el algoritmo SAMF y para el global de los algoritmos, este es poco apreciable. Sin embargo, en el caso de los cambios de aspecto esta diferencia ya es notable, produciéndose en los algoritmos CT y SAMF y en la media global de todos los algoritmos.

Por último cabe destacar la gran mejora obtenida para los retos de camuflaje y de oclusiones. Dentro de las secuencias evaluadas las mejoras más notables se han dado en las secuencias de camuflaje, con colores llamativos y con colores oscuros, y en las oclusiones, siendo las últimas las que muestran tanto la mejoría más grande como la diferencia absoluta más abultada entre los resultados de las secuencias corregidas y las secuencias sin corregir. Por lo tanto se puede afirmar que el reto de las oclusiones es el que más favorecido se ha visto por la introducción de la corrección del *jitter*.

Capítulo 6

Conclusiones y trabajo futuro.

6.1. Conclusiones.

El objetivo que se pretendía cumplimentar en este proyecto consistía en lograr acoplar algoritmos orientados al uso en cámaras fijas en cámaras móviles para potenciar, de este modo, las ventajas heredadas de usar una cámara móvil. Éste se ha logrado de forma satisfactoria, obteniendo unos resultados mejores de los que a priori se podían esperar.

Para la consecución de dicho objetivo, fue necesario en primer lugar modelar teóricamente los efectos y retos a los que se enfrentaba el proyecto, tanto desde el punto de vista de los algoritmos utilizados, retos de seguimiento o *tracking*, como desde el punto de vista de los problemas que acarrea el uso de cámaras móviles, retos de los UAVs. Tras el estudio, se puede concluir que los retos de los UAVs de mayor complejidad son los de vibración de la cámara y desenfoque de la imagen por el movimiento. Su combinación con retos de seguimiento como las oclusiones o los movimientos bruscos también incluyen gran complejidad.

Una vez modelados los problemas, era necesario evaluar los efectos de estos retos sobre algoritmos de seguimiento del estado del arte. Para ello se diseñó un conjunto de datos sencillo, sintético, y muy orientado a la problemática enfrentada, que permitió sacar las siguientes conclusiones. En primer lugar, fue posible hacer una selección de aquellos algoritmos, dos CT y SAMF, que mejor funcionamiento presentaban en estos entornos. En segundo lugar, los resultados permitieron diseñar las propuestas de mejora presentadas.

A partir de los análisis previos fue posible proponer una serie de mejoras para adaptar el funcionamiento de los algoritmos escogidos sobre secuencias capturadas desde UAVs. Dichas mejoras fueron orientadas a adaptar las condiciones de captura, de

tal manera que la secuencia sobre la que trabaje el algoritmo de seguimiento sea lo más similar a una capturada desde una cámara estática. Los algoritmos de corrección del emborronado y la estabilización de cámara han permitido mejorar notablemente las condiciones de trabajo de los algoritmos de seguimiento, pero a costa de un incremento leve del coste computacional.

Para la evaluación del sistema completo se generó un segundo conjunto de datos. Este conjunto de datos ya presentaba condiciones reales, esto es, fue grabado desde un UAV. En el conjunto grabado se incorporaron secuencias de seguimiento con todo tipo de eventos, como cambios de escala, de apariencia o de dirección del movimiento. Sobre ese conjunto se realizó una evaluación comparativa entre los algoritmos de cámara fija escogidos, y dichos algoritmos adaptados.

De los resultados finales se pueden extraer dos conclusiones. La primera, que en base a la mejora observada, las propuestas satisfacen el objetivo inicial del proyecto. La segunda, relacionada con las situaciones en las que la mejora no siempre se ve reflejada. En dichas situaciones, se puede concluir que la modificación introducida garantiza buen funcionamiento cuando trabaja como debe, sin embargo, cuando la estabilización de la imagen falla los resultados se ven notablemente afectados.

6.2. Trabajo futuro.

Una vez finalizado este proyecto aparecen tres líneas diferentes sobre las que trabajar en un futuro con el objetivo de mejorar o progresar a partir de lo logrado.Las que se han considerado más destacadas, y que se detallarán a continuación serían: ampliar el conjunto de datos sobre el que trabajar y añadir mejoras a nivel de la propuesta de mejora.

En cuanto a la ampliación del conjunto de datos se proponen varias adiciones. La primera consistiría en incrementar el mismo haciendo uso de secuencias de vídeo más largas en las que puedan aparecer varios retos combinados. De este modo se podría hacer una evaluación estadísticamente más representativa del funcionamiento de las mejoras introducidas. También se propone añadir secuencias en diferentes entornos, con el fin de evaluar la dependencia que puedan tener los resultados del entorno de grabación. Añadir diferentes vistas de las mismas secuencias podría aportar información acerca de la importancia de grabar a cierta distancia y altura, pudiendo optimizar dicha distancia y altura dependiendo de los resultados obtenidos. Otra adición que se puede hacer al conjunto de datos que se ha usado en este proyecto es el de incluir secuencias en las que las condiciones de captura sean peores. Esto incluye trabajar en condiciones poco propicias para el dron. Un fuerte viento desestabiliza al UAV

haciendo más complicada si cabe la tarea de seguimiento. Otra de las condiciones en la que se podrían grabar algunas secuencias es en condiciones de lluvia. En caso de disponer de un UAV y una cámara resistentes a la lluvia resulta muy útil grabar en estas condiciones debido a que la lluvia supone un reto más en la escena, dificultando el seguimiento del objetivo incluso en cámaras fijas.

Relacionado con cambios a nivel de propuestas de mejora, estos consisten sobre todo en mejorar las propuestas desarrolladas. El primer cambio podría ir orientado a montar de algún modo las correcciones de las secuencias sobre el propio UAV, pudiendo corregir las secuencias en tiempo real para evaluarlas. El segundo cambio consistiría en incorporar en dicha etapa de corrección una etapa de rellenado de los bordes de las imágenes más realista que el implementado en este proyecto, debido a que en condiciones en las que la estabilidad del UAV sea reducida éste efecto se volverá más relevante.

Por último, aunque ya con más trabajo a realizar, un objetivo futuro de este proyecto podría ser la interacción del algoritmo de seguimiento con el propio aparato, de tal manera que se pudiera garantizar el seguimiento automático del objetivo.

Bibliografía

[1] F. LIRIS, "The visual object tracking vot2014 challenge results,"

[2] K. L. Cook, "The silent force multiplier: the history and role of uavs in warfare," in *2007 IEEE Aerospace Conference*, pp. 1–7, IEEE, 2007.

[3] D. Erdos and S. E. Watkins, "Uav autopilot integration and testing," in *Region 5 Conference, 2008 IEEE*, pp. 1–6, IEEE, 2008.

[4] A. M. Samad, N. Kamarulzaman, M. A. Hamdani, T. A. Mastor, and K. A. Hashim, "The potential of unmanned aerial vehicle (uav) for civilian and mapping application," in *System Engineering and Technology (ICSET), 2013 IEEE 3rd International Conference on*, pp. 313–318, IEEE, 2013.

[5] H.-J. Schwaerzler, "Method for remote-controlling an unmanned aerial vehicle," Apr. 23 2002. US Patent 6,377,875.

[6] H. Chao, Y. Cao, and Y. Chen, "Autopilots for small unmanned aerial vehicles: a survey," *International Journal of Control, Automation and Systems*, vol. 8, no. 1, pp. 36–44, 2010.

[7] Y. Watanabe, A. J. Calise, and E. N. Johnson, "Vision-based obstacle avoidance for uavs," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, vol. 11, 2007.

[8] A. Khvilivitzky, "Visual collision avoidance system for unmanned aerial vehicles," Dec. 3 1996. US Patent 5,581,250.

[9] R. Carnie, R. Walker, and P. Corke, "Image processing algorithms for uav"sense and avoid"," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 2848–2853, IEEE, 2006.

[10] A. Ryan, M. Zennaro, A. Howell, R. Sengupta, and J. K. Hedrick, "An overview of emerging results in cooperative uav control," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 1, pp. 602–607, IEEE, 2004.

[11] X. Wang, V. Yadav, and S. Balakrishnan, "Cooperative uav formation flying with obstacle/collision avoidance," *IEEE Transactions on control systems technology*, vol. 15, no. 4, pp. 672–679, 2007.

[12] Y. M. Chen, L. Dong, and J.-S. Oh, "Real-time video relay for uav traffic surveillance systems through available communication networks," in *2007 IEEE Wireless Communications and Networking Conference*, pp. 2608–2612, IEEE, 2007.

[13] R. Mori, K. Hirata, and T. Kinoshita, "Vision-based guidance control of a small-scale unmanned helicopter," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2648–2653, IEEE, 2007.

[14] G. R. Rodríguez-Canosa, S. Thomas, J. del Cerro, A. Barrientos, and B. MacDonald, "A real-time method to detect and track moving objects (datmo) from unmanned aerial vehicles (uavs) using a single camera," *Remote Sensing*, vol. 4, no. 4, pp. 1090–1111, 2012.

[15] A. Viquerat, L. Blackhall, A. Reid, S. Sukkarieh, and G. Brooker, "Reactive collision avoidance for unmanned aerial vehicles using doppler radar," in *Field and Service Robotics*, pp. 245–254, Springer, 2008.

[16] W. K. Bodin, J. J. Redman, and D. C. Thorson, "Navigating a uav under remote control and manual control with three dimensional flight depiction," Feb. 15 2005. US Patent 6,856,894.

[17] B. Coifman, M. McCord, M. Mishalani, and K. Redmill, "Surface transportation surveillance from unmanned aerial vehicles," in *Proc. of the 83rd Annual Meeting of the Transportation Research Board*, 2004.

[18] S. Srinivasan, H. Latchman, J. Shea, T. Wong, and J. McNair, "Airborne traffic surveillance systems: video surveillance of highway traffic," in *Proceedings of the ACM 2nd international workshop on Video surveillance & sensor networks*, pp. 131–135, ACM, 2004.

[19] A. Ollero and L. Merino, "Unmanned aerial vehicles as tools for forest-fire fighting," *Forest Ecology and Management*, vol. 234, no. 1, p. S263, 2006.

[20] D. W. Casbeer, R. W. Beard, T. W. McLain, S.-M. Li, and R. K. Mehra, "Forest fire monitoring with multiple small uavs," in *Proceedings of the 2005, American Control Conference, 2005.*, pp. 3530–3535, IEEE, 2005.

[21] L. Merino, F. Caballero, J. R. Martínez-de Dios, I. Maza, and A. Ollero, "An unmanned aircraft system for automatic forest fire monitoring and measurement," *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1-4, pp. 533–548, 2012.

[22] T.-Y. Chou, M.-L. Yeh, Y. C. Chen, and Y. H. Chen, *Disaster monitoring and management by the unmanned aerial vehicle technology.* na, 2010.

[23] S. M. Adams and C. J. Friedland, *A survey of unmanned aerial vehicle (UAV) usage for imagery collection in disaster research and management.* publisher not identified, 2011.

[24] M. A. Goodrich, L. Lin, B. S. Morse, and M. Roscheck, "Supporting wilderness search and rescue with integrated intelligence: autonomy and information at the right time and the right place," AAAI, 2010.

[25] R. Heß, M. Fritscher, M. Krauß, and K. Schilling, "Setting up a surveillance system in the civil domain with cooperating uavs and ugvs," *IFAC Proceedings Volumes*, vol. 45, no. 28, pp. 19–24, 2012.

[26] E. Bone and C. Bolkcom, "Unmanned aerial vehicles: Background and issues for congress," DTIC Document, 2003.

[27] J. Kim and Y. Kim, "Moving ground target tracking in dense obstacle areas using uavs," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 8552–8557, 2008.

[28] M. Quigley, M. A. Goodrich, S. Griffiths, A. Eldredge, and R. W. Beard, "Target acquisition, localization, and surveillance using a fixed-wing mini-uav and gimbaled camera," in *Proceedings of the 2005 IEEE international conference on robotics and automation*, pp. 2600–2605, IEEE, 2005.

[29] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, 2014.

[30] E. Maggio and A. Cavallaro, *Video tracking: theory and practice*. John Wiley & Sons, 2011.

[31] F. Navarro, "Estudio de viabilidad de aplicaciones de video-vigilancia en cámaras exteriores.." Diciembre 2014.

[32] D. Koppel, Y.-F. Wang, and H. Lee, "Robust and real-time image stabilization and rectification," in *Application of Computer Vision, 2005. WACV/MOTIONS'05 Volume 1. Seventh IEEE Workshops on*, vol. 1, pp. 350–355, IEEE, 2005.

[33] Y. Mai, H. Zhao, and S. Guo, "The analysis of image stabilization technology based on small-uav airborne video," in *Proceedings of International Conference on Computer Science & Electronics Engineering, Hangzhou, China*, vol. 2325, p. 586589, 2012.

[34] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, 2011.

[35] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, 2015.

[36] D. P. Young and J. M. Ferryman, "Pets metrics: On-line performance evaluation service," in *Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance (VS-PETS)*, pp. 317–324, 2005.

[37] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2411–2418, 2013.

[38] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, F. Porikli, L. Cehovin, G. Nebehay, G. Fernandez, T. Vojir, A. Gatt, *et al.*, "The visual object tracking vot2013 challenge results," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 98–111, 2013.

[39] B. Karasulu and S. Korukoglu, "A software for performance evaluation and comparison of people detection and tracking methods in video processing," *Multimedia Tools and Applications*, vol. 55, no. 3, pp. 677–723, 2011.

[40] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, no. 1, pp. 1–10, 2008.

[41] A. Polesel, G. Ramponi, V. J. Mathews, *et al.*, "Image enhancement via adaptive unsharp masking," *IEEE transactions on image processing*, vol. 9, no. 3, pp. 505–510, 2000.

[42] E. Rosten and T. Drummond, "Fusing points and lines for high performance tracking," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2, pp. 1508–1515, IEEE, 2005.

[43] A. Alahi, R. Ortiz, and P. Vandergheynst, "Freak: Fast retina keypoint," in *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*, pp. 510–517, Ieee, 2012.

[44] M. Muja and D. G. Lowe, "Fast matching of binary features," in *Computer and Robot Vision (CRV), 2012 Ninth Conference on*, pp. 404–410, IEEE, 2012.

[45] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[46] S. Wang, H. Lu, F. Yang, and M.-H. Yang, "Superpixel tracking," in *2011 International Conference on Computer Vision*, pp. 1323–1330, IEEE, 2011.

[47] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference, Nottingham, September 1-5, 2014*, BMVA Press, 2014.

[48] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 2544–2550, IEEE, 2010.

[49] Y. Li and J. Zhu, "A scale adaptive kernel correlation filter tracker with feature integration," in *European Conference on Computer Vision*, pp. 254–265, Springer, 2014.

[50] M. Danelljan, F. Shahbaz Khan, M. Felsberg, and J. Van de Weijer, "Adaptive color attributes for real-time visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1090–1097, 2014.

[51] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *European conference on computer vision*, pp. 702–715, Springer, 2012.

[52] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

[53] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 125–141, 2008.

[54] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *European Conference on Computer Vision*, pp. 864–877, Springer, 2012.

[55] K. Briechle and U. D. Hanebeck, "Template matching using fast normalized cross correlation," in *Aerospace/Defense Sensing, Simulation, and Controls*, pp. 95–102, International Society for Optics and Photonics, 2001.

[56] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *International journal of computer vision*, vol. 56, no. 3, pp. 221–255, 2004.

Apéndice A

Análisis teórico y práctico de algoritmos de tracking

Discriminative Scale Space Tracker (DSST)

M. Danelljan, *G. Häger*, *F. S. Khan*, *M. Felsberg* (fmartin.danelljan@liu.se, hager.gustav@gmail.com, ffahad.khan, michael.felsbergg@liu.se)

DSST [47] es una extensión del algoritmo Minimum Output Sum of Squared Errors (MOSSE) [48] que incluye una estimación robusta de escala del objetivo. MOSSE entrena un filtro de correlación discriminativo sobre una serie de muestras de parches en escala de grises para emplearlo en la estimación del traslado del objetivo en el próximo *frame*. A su vez DSST hace uso de un filtro de escala discriminativo unidimensional con el fin de estimar el tamaño del objetivo. Como podemos observar en la Figura A.1 DSST funciona muy bien con oclusiones, cambios de iluminación y movimiento de la cámara, sin embargo no es tan preciso en cambios de escala a pesar del filtro de escala empleado.

A Kernel Correlation Filter Tracker with Scale Adaptive and Feature Integration (SAMF)

Y. Li, *J. Zhu* (fliyang89, jkzhug@zju.edu.cn)

SAMF [49] es un algoritmo de *tracking* que tiene su origen en los *trackers* basados en filtros de correlación [48, 50, 51, 52] con el objetivo de mejorar la capacidad de seguimiento. Para abordar los problemas que tienen estos *trackers* con los cambios de escala, SAMF propone emplear un esquema de escala adaptivo, mejorando de este modo las prestaciones ofrecidas por los *trackers* basados en filtros de correlación, los cuales hacen uso de plantillas de un tamaño fijo.

Además de esta ventaja, SAMF también se ayuda de características como HoG y el color para elevar su rendimiento en términos de seguimiento de objetivos. Este algoritmo tiene buenos resultados para movimientos de la cámara, cambios de iluminación y cambios de movimiento. Sin embargo, no es muy robusto para los cambios de escala y no funciona tan bien para oclusiones, aún siendo de los mejores en este ámbito.

Kernelized Correlation Filter Tracker (KCF)

J. F. Henriques, *J. Batista* (fhenriques, batistag@isr.uc.pt)

Se trata de un filtro de correlación [52] que opera sobre características HoG. Las mejoras respecto a la versión anterior son soporte de múltiples escalas, estimación de picos por bloques o celdas y reemplazo de las actualizaciones del modelo mediante interpolación lineal con un esquema de actualización más robusto [50]. A tenor de los resultados mostrados en la Figura A.1, el mayor problema de KCF está relacionado con la robustez del algoritmo. En términos de exactitud muestra buenos resultados para cambios de escala, de iluminación, de movimiento, oclusiones y movimiento de la cámara, sin embargo tanto en oclusiones como en cambios de escala tiene problemas a la hora de no perder nunca el objetivo.

Incremental Learning for Robust Visual Tracking (IVT)

VOT 2014 *TechnicalCommittee*

El *tracker* IVT [53] aprende de forma incremental una representación del subespacio en pocas dimensiones, adaptándose durante el seguimiento a los posibles cambios en la apariencia del objetivo. La actualización del modelo está basada en algoritmos incrementales para el análisis de las componentes principales e incluye dos características: un método para actualizar de forma correcta la media de las muestras, y un factor que nos asegura que gastamos menos recursos en el modelado. El mejor rendimiento de IVT se da en los cambios de iluminación en términos de exactitud a la hora de seguir un objetivo, no obstante es un algoritmo que no da buenos resultados ni en exactitud ni en robustez para ninguno de los retos que aparecen en la Figura A.1.

Compressive Tracking (CT)

VOT 2014 *Technical Committee*

El algoritmo CT [54] emplea un modelo de apariencia basado en las características extraídas del espacio de características de imagen multi-escala. Hace uso de

proyecciones aleatorias no adaptativas que preservan la estructura del espacio de características de imagen de los objetos. Para extraer dichas características del modelo de apariencia de forma eficiente utiliza una matriz de medida muy dispersa. Esta misma matriz es la encargada de comprimir las muestras del fondo y del primer plano. La tarea de *tracking* se formula como una clasificación binaria a partir de un clasificador sencillo de Bayes con actualizaciones llevadas a cabo de forma *online* en el dominio de la compresión. Donde mejores resultados tiene el algoritmo CT es en las oclusiones, en el resto de retos presentados en la Figura A.1 el rendimiento de CT es pobre tanto en términos de robustez como en exactitud en el seguimiento. Los resultados a priori son malos.

Normalized Cross-Correlation (NCC)

K. Briechle, *U. D. Hanebeck*

NCC [55] emplea los valores de intensidad del *bounding box* en el *frame* inicial como plantilla para estimar el movimiento del objetivo. Esta plantilla no será actualizada. Para cada *frame* el algoritmo realiza una búsqueda uniforme alrededor de la posición del objetivo en el anterior *frame* mediante ventanas de muestra. Cada de una de las ventanas se compara con la plantilla inicial, haciendo uso de la correlación cruzada normalizada. La ventana que obtiene la mayor puntuación representa la posición del objetivo en ese *frame*.

Lucas-Kanade Tracker (KLT)

S. Baker, *I. Matthews* (simonb@cs.cmu.edu, iainm@cs.cmu.edu)

Este algoritmo se propuso por primera vez en el 1981, siendo de este modo uno de los pioneros dentro del *tracking*. KLT [56] hace uso de una transformada afín para encontrar coincidencias entre el *bounding box* del objetivo y ventanas alrededor de su localización anterior. Dicha transformada afín es calculada mediante alineamiento de imagen incremental en derivadas espacio-temporales y *warping* [46] capaz de lidiar con cambios de escala, rotación y traslación. La posición del objetivo se obtiene mapeando la posición del objetivo en el *frame* anterior con la localización del mismo en el *frame* a estimar haciendo uso de la transformada afín calculada.

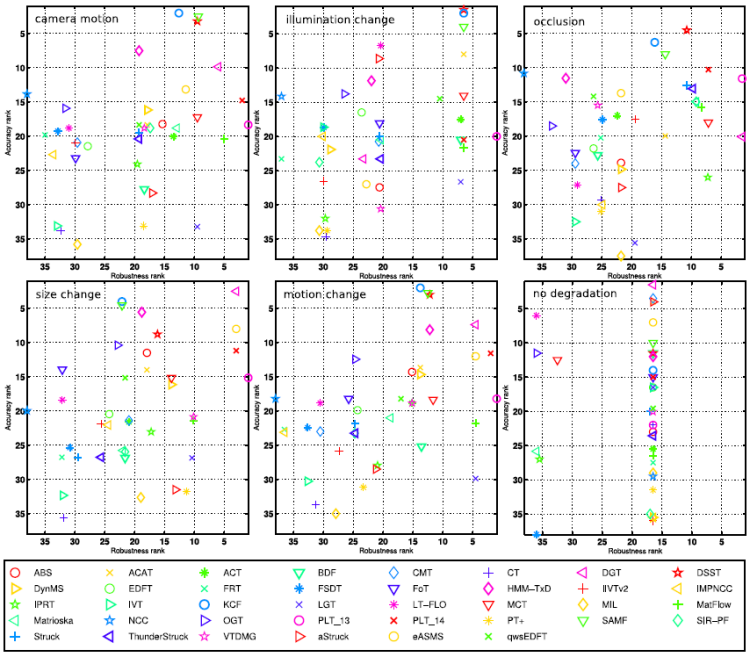


Figura A.1: Comparación entre los algoritmos del VOT para retos indicados (esquina superior izquierda). Fuente [1].

Apéndice B

Resultados del toy example.

Seguidamente se mostrarán los resultados en términos de *F1-score* obtenidos por todos los algoritmos para las secuencias grabadas en el *toy example*, las secuencias sintéticas y las secuencias con *jitter* corregido.

B.1. CT.

Retos	SecOrig	SecCamIlu	SecRuiAle	Sec*Jitter*	SecCorreg
Bounc_Mov_Lento	0,8597	0,8005	0,8085	0,8273	0,8311
Bounc_Mov_Rapido	0,8023	0,7411	0,8081	0,7432	0,7523
Cambio_Apariencia	0,5233	0,7461	0,7813	0,7102	0,8292
Cambio_Aspecto	0,8474	0,7589	0,8361	0,7161	0,7953
Camuflaje	0,7927	0,7916	0,7682	0,8337	0,6798
Oclusiones	0,7591	0,4902	0,6147	0,7497	0,4834
Paseo_Simple	0,8253	0,7657	0,8359	0,7960	0,8276

Tabla B.1: Puntuaciones obtenidas por CT para las secuencias grabadas, generadas y corregidas del *toy example*.

B.2. DSST.

Retos	SecOrig	SecCamIlu	SecRuiAle	Sec*Jitter*	SecCorreg
Bounc_Mov_Lento	0,7783	0,8075	0,7747	0,7991	0,7954
Bounc_Mov_Rapido	0,7582	0,4891	0,7519	0,7734	0,4060
Cambio_Apariencia	0,8336	0,7661	0,8074	0,6383	0,6133
Cambio_Aspecto	0,7209	0,6984	0,7534	0,7279	0,8631
Camuflaje	0,8150	0,7894	0,6800	0,7616	0,6146
Oclusiones	0,7309	0,2709	0,7435	0,2262	0,7735
Paseo_Simple	0,7902	0,8036	0,8254	0,6285	0,8251

Tabla B.2: Puntuaciones obtenidas por DSST para las secuencias grabadas, generadas y corregidas del *toy example*.

B.3. IVT

Retos	SecOrig	SecCamIlu	SecRuiAle	Sec*Jitter*	SecCorreg
Bounc_Mov_Lento	0,2928	0,1416	0,1554	0,1785	0,5576
Bounc_Mov_Rapido	0,0848	0,0911	0,0579	0,0696	0,0718
Cambio_Apariencia	0,1880	0,2888	0,1395	0,1374	0,2600
Cambio_Aspecto	0,3581	0,3295	0,2698	0,2052	0,1445
Camuflaje	0,2886	0,3661	0,2549	0,0234	0,2302
Oclusiones	0,2752	0,2798	0,0864	0,0688	0,1463
Paseo_Simple	0,7139	0,7846	0,3340	0,2100	0,0551

Tabla B.3: Puntuaciones obtenidas por IVT para las secuencias grabadas, generadas y corregidas del *toy example*.

B.4. KCF

Retos	SecOrig	SecCamIlu	SecRuiAle	Sec*Jitter*	SecCorreg
Bounc_Mov_Lento	0,8328	0,8324	0,8129	0,8036	0,7108
Bounc_Mov_Rapido	0,4430	0,5084	0,7294	0,4880	0,4375
Cambio_Apariencia	0,5228	0,5243	0,7892	0,4591	0,3853
Cambio_Aspecto	0,0771	0,8078	0,7627	0,7953	0,9009
Camuflaje	0,8081	0,7334	0,8368	0,7560	0,7498
Oclusiones	0,7625	0,5499	0,7393	0,7566	0,6226
Paseo_Simple	0,8224	0,8484	0,8332	0,6941	0,8635

Tabla B.4: Puntuaciones obtenidas por KCF para las secuencias grabadas, generadas y corregidas del *toy example*.

B.5. KLT

Retos	SecOrig	SecCamIlu	SecRuiAle	Sec*Jitter*	SecCorreg
Bounc_Mov_Lento	0,2668	0,2346	0,3734	0,1975	0,1073
Bounc_Mov_Rapido	0,0968	0,0996	0,1051	0,0997	0,0486
Cambio_Apariencia	0,2061	0,0865	0,0721	0,2025	0,0843
Cambio_Aspecto	0,4231	0,2758	0,4378	0,4389	0,2748
Camuflaje	0,0271	0,0270	0,2632	0,0284	0,0138
Oclusiones	0,2420	0,2803	0,2845	0,2446	0,1509
Paseo_Simple	0,1169	0,0877	0,1860	0,0964	0,0607

Tabla B.5: Puntuaciones obtenidas por KLT para las secuencias grabadas, generadas y corregidas del *toy example*.

B.6. NCC

Retos	SecOrig	SecCamIlu	SecRuiAle	Sec*Jitter*	SecCorreg
Bounc_Mov_Lento	0,1289	0,1288	0,0560	0,0466	0,0648
Bounc_Mov_Rapido	0,0248	0,0248	0,0407	0,0069	0,0034
Cambio_Apariencia	0,0838	0,0838	0,0351	0,0082	0,0057
Cambio_Aspecto	0,1849	0,1482	0,0796	0,0053	0,0105
Camuflaje	0,0329	0,0329	0,0321	0,0036	0,0040
Oclusiones	0,0148	0,0148	0,0184	0,0042	0,0139
Paseo_Simple	0,0345	0,0344	0,1128	0,0085	0,1863

Tabla B.6: Puntuaciones obtenidas por NCC para las secuencias grabadas, generadas y corregidas del *toy example*.

B.7. SAMF

Retos	SecOrig	SecCamIlu	SecRuiAle	Sec*Jitter*	SecCorreg
Bounc_Mov_Lento	0,8149	0,8184	0,8150	0,5806	0,8377
Bounc_Mov_Rapido	0,4960	0,5576	0,5090	0,6897	0,4536
Cambio_Apariencia	0,7949	0,5154	0,7647	0,4585	0,6662
Cambio_Aspecto	0,7896	0,8082	0,7729	0,7746	0,8633
Camuflaje	0,8459	0,7692	0,7742	0,7690	0,6088
Oclusiones	0,7226	0,7024	0,2430	0,7169	0,2701
Paseo_Simple	0,7997	0,7670	0,8277	0,6999	0,8348

Tabla B.7: Puntuaciones obtenidas por SAMF para las secuencias grabadas, generadas y corregidas del *toy example*.

Apéndice C

Resultados de las secuencias reales.

A continuación se mostrarán en varias tablas los resultados obtenidos por todos los algoritmos tanto para las secuencias originales como para las secuencias corregidas.

C.1. CT

Retos	SecOrig	SecCorreg
Bounc_Mov_Lento	0,9212	0,9278
Bounc_Mov_Rapido	0,7801	0,6901
Cambio_Apariencia	0,8256	0,8524
Cambio_Aspecto	0,7947	0,7129
Camuflaje_Soft_Color	0,8758	0,9204
Camuflaje_Strong_Color	0,7554	0,8975
Oclusiones	0,8720	0,8860

Tabla C.1: Puntuaciones obtenidas por CT para las secuencias reales grabadas y corregidas.

C.2. DSST

Retos	SecOrig	SecCorreg
Bounc_Mov_Lento	0,8935	0,9185
Bounc_Mov_Rapido	0,6689	0,6464
Cambio_Apariencia	0,8836	0,9255
Cambio_Aspecto	0,8318	0,7603
Camuflaje_Soft_Color	0,8578	0,8711
Camuflaje_Strong_Color	0,8338	0,9014
Oclusiones	0,8718	0,9387

Tabla C.2: Puntuaciones obtenidas por DSST para las secuencias reales grabadas y corregidas.

C.3. IVT

Retos	SecOrig	SecCorreg
Bounc_Mov_Lento	0,0421	0,0637
Bounc_Mov_Rapido	0,0852	0,0177
Cambio_Apariencia	0,0634	0,0647
Cambio_Aspecto	0,1598	0,0411
Camuflaje_Soft_Color	0,0687	0,0831
Camuflaje_Strong_Color	0,0526	0,0453
Oclusiones	0,0758	0,3125

Tabla C.3: Puntuaciones obtenidas por IVT para las secuencias reales grabadas y corregidas.

C.4. KCF

Retos	SecOrig	SecCorreg
Bounc_Mov_Lento	0,9098	0,8881
Bounc_Mov_Rapido	0,6686	0,5530
Cambio_Apariencia	0,9429	0,9253
Cambio_Aspecto	0,8391	0,7359
Camuflaje_Soft_Color	0,9138	0,9501
Camuflaje_Strong_Color	0,8592	0,9175
Oclusiones	0,8619	0,9351

Tabla C.4: Puntuaciones obtenidas por KCF para las secuencias reales grabadas y corregidas.

C.5. KLT

Retos	SecOrig	SecCorreg
Bounc_Mov_Lento	0,0980	0,1064
Bounc_Mov_Rapido	0,3631	0,1077
Cambio_Apariencia	0,0748	0,0143
Cambio_Aspecto	0,3214	0,4824
Camuflaje_Soft_Color	0,1805	0,1701
Camuflaje_Strong_Color	0,1976	0,0157
Oclusiones	0,0321	0,5976

Tabla C.5: Puntuaciones obtenidas por KLT para las secuencias reales grabadas y corregidas.

C.6. NCC

Retos	SecOrig	SecCorreg
Bounc_Mov_Lento	0,1220	0,1002
Bounc_Mov_Rapido	0,0580	0,0322
Cambio_Apariencia	0,0839	0,0264
Cambio_Aspecto	0,0376	0,0460
Camuflaje_Soft_Color	0,1187	0,1684
Camuflaje_Strong_Color	0,2229	0,2233
Oclusiones	0,1272	0,0606

Tabla C.6: Puntuaciones obtenidas por NCC para las secuencias reales grabadas y corregidas.

C.7. SAMF

Retos	SecOrig	SecCorreg
Bounc_Mov_Lento	0,9110	0,9094
Bounc_Mov_Rapido	0,6841	0,7634
Cambio_Apariencia	0,9424	0,9284
Cambio_Aspecto	0,8144	0,7558
Camuflaje_Soft_Color	0,8865	0,9377
Camuflaje_Strong_Color	0,8425	0,9177
Oclusiones	0,8767	0,9387

Tabla C.7: Puntuaciones obtenidas por SAMF para las secuencias reales grabadas y corregidas.